UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Controle de Formação de Robôs Móveis Baseado em Visão Omnidirecional

Christiano Couto Gava

Vitória

Agosto de 2007

Christiano Couto Gava

Controle de Formação de Robôs Móveis Baseado em Visão Omnidirecional

Universidade Federal do Espírito Santo
Centro Tecnológico
Programa de Pós-Graduação em Engenharia Elétrica

Vitória

Agosto de 2007

.

Dados Internacionais de Catalogação-na-publicação (CIP)
(Biblioteca Central da Universidade Federal do Espírito Santo, ES, Brasil)

Gava, Christiano Couto, 1979-
G279c Controle de Formação de Robôs Móveis Baseado em Visão Omnidirecional / Christiano Couto Gava. - 2007.
104 f. : il.

Orientadora: Raquel Frizera Vassallo.
Co-Orientador: Teodiano Freire Bastos Filho.
Dissertação (mestrado) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Robótica. 2. Processamento de imagens. I. Vassallo, Raquel Frizera. II. Bastos Filho, Teodiano Freire. III. Universidade Federal do Espírito Santo. Centro Tecnológico. IV. Título.

CDU: 621.3

.

Christiano Couto Gava

Controle de Formação de Robôs Móveis Baseado em Visão Omnidirecional

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica – Automação.

Aprovada em 10 de agosto de 2007.

Comissão Examinadora:

Prof[a]. Dr[a].Raquel Frizera Vassallo
Universidade Federal do Espírito Santo, Orientadora

Prof. Dr. Teodiano Freire Bastos Filho
Universidade Federal do Espírito Santo, Co-orientador

Prof. Dr. Carlos Henrique Costa Ribeiro
Instituto Tecnológico de Aeronáutica

Prof. Dr. Ricardo Carelli
Instituto de Automática - Universidade Nacional de San Juan - Argentina

Vitória, agosto de 2007.

Dedicatória

Aos meus avós, Waldemiro e Nascípio, homens de bem e exemplos de força e determinação.

Agradecimentos

Eu preciso escrever um texto de Agradecimentos.

Prefiro sair por aí e agradecer pessoalmente a cada um, mas eu tenho que escrever um texto.

Escrever é difícil. Pelo menos para mim.

Gostaria de ter o dom da escrita, como o têm Veríssimo, Jabor e outros. Assim, poderia expressar de forma clara (e, quem sabe, divertida) a minha gratidão. Mas não tenho esse dom. Portanto, vou me contentar em escrever um texto simples e direto.

Agradeço, em primeiro lugar, aos meus pais, Alcemir e Anacir, minha irmã, Juliana, e minha namorada, Patrícia, pelo apoio e pela compreensão desde a época da Graduação. Sem vocês, esse trabalho sequer teria começado.

À Prof^{a}. Dr^{a}. Raquel Frizera Vassallo que, como todos sabem, é muito mais que uma orientadora. É orientadora, amiga, conselheira, coordenadora do *LAI Expedition*, jogadora de vôlei (excelente qualidade!), puxadora de orelha (!) e detentora de várias outras funções legais. Sem dúvida, Raquel, você é uma pessoa iluminada. Nem vou tentar escrever um parágrafo grande para te agradecer... Não seria o suficiente mesmo!

Ao Prof. Dr. Teodiano Freire Bastos Filho, co-orientador, por toda boa vontade e atenção dispensadas. Suas dicas e sugestões, recheados de sabedoria e experiência, ajudaram bastante!

Aos professores do Programa de Pós-Graduação da Engenharia Elétrica, pelos ensinamentos e conselhos que, sem dúvida, foram fundamentais na realização desse trabalho.

Aos professores, estudantes e funcionários do Instituto de Automática (INAUT) da Universidade Nacional de San Juan, Argentina, pelo excelente tratamento dado a nós, brasileiros. Agradeço em especial ao Prof. Dr. Ing. Ricardo Carelli e ao doutorando Flavio Roberti, pelos ensinamentos em Controle Não Linear e por permitirem a utilização e adaptação do controlador de formação por eles idealizado neste trabalho. Não posso deixar de agradecer também em especial ao Prof. Dr. Ing. Carlos Soria (Carlitos), por compartilhar sua casa com André Ferreira e eu durante nossa estadia em San Juan.

Aos amigos do LAI, como Flávio, sempre disposto a ajudar, fazer bagunça e jogar vôlei; e

Rodrigo Rosenfeld, com dicas importantes sobre programação. A Fabrício (Frodo) e Marino, que, além de todo o suporte na montagem dos robôs, desempenharam com excelência os papéis de *camera-man* e *cable-man* durante a realização dos experimentos finais. Muito obrigado pela ajuda naquele sábado. Muito obrigado também a Wanderley, Daniel, Alexandre Konzen, Alexandre (Timótiu), André Ferreira, Mariana, Rafael Leal (PS!), Luismar e todos aqueles que não só compreenderam, mas também ajudaram na realização dos experimentos preliminares. Roger, Tia Sandra, Andrezinho, Lorena, Lester e Franco, não esqueci de vocês. E é claro, Felipe Martins, que surpreendeu a todos com seu senso de humor "apurado", na estória do "Grande Monstro Verde Comedor de Pedra". Essa foi ótima!

Obrigado também ao Dr. Ing. Celso De La Cruz Casaño, pela grande cooperação no trabalho realizado em San Juan.

Obrigado ao Prof. Dr. Paulo Faria Santos Amaral, que, através da Automática, ajudou na montagem dos robôs.

À CAPES, pela bolsa de estudo e pelo financiamento do meu estágio em San Juan.

Sandro, Léo e Valmor: agora sim vocês podem me chamar de Mestre!

Posso mandar um abraço no texto de Agradecimentos? Sei lá! De qualquer forma, um grande abraço a todos vocês e MUITO OBRIGADO!

Christiano Couto Gava

“Aprendi que um homem só tem o direito de olhar um outro de cima para baixo para ajudá-lo a levantar-se.”

Gabriel Garcia Marquez

Lista de Tabelas

Lista de Figuras

Sumário

Resumo

Este trabalho aborda o problema do controle de formação de uma equipe de robôs móveis. O controle adotado é do tipo centralizado e foi projetado segundo a teoria de controle não linear. A equipe de robôs é formada por um líder, que possui maior capacidade de processamento, e de outros robôs mais simples e baratos, chamados de seguidores. O líder é o responsável por conduzir toda a equipe à formação desejada durante a navegação. Para isso, é equipado com um sistema de visão omnidirecional, que permite estimar a posição e a orientação de todos os seguidores capturando apenas uma imagem. A realimentação do controlador, totalmente baseada em visão, é obtida através de técnicas de processamento digital de imagens. Os algoritmos utilizados para processar as imagens omnidirecionais contribuem para o aumento da robustez e do desempenho do sistema.

Simulações foram feitas para estudar o comportamento do controlador diante de várias formações diferentes. Os resultados obtidos nos experimentos com robôs reais comprovaram o bom desempenho do sistema de controle proposto.

Abstract

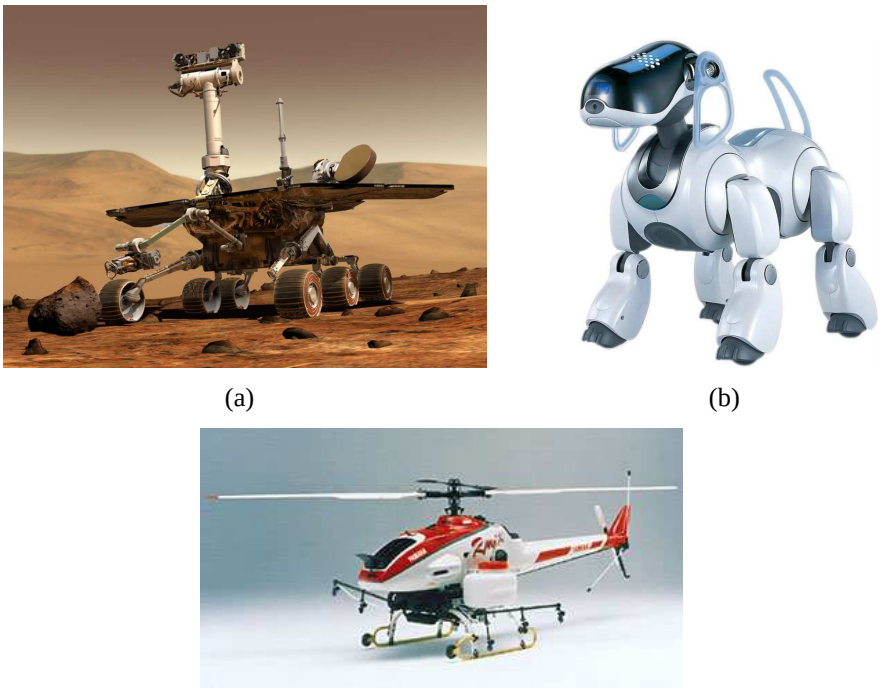
This work addresses the problem of controlling the formation of a group of mobile robots. A centralized control was chosen and projected according to nonlinear control theory. The team is composed by a leader with major processing capacity, and other simple and cheap robots, called followers. The leader is responsible for driving the group to the desired formation during navigation and is equiped with an omnidirectional vision system in order to estimate the followers position and orientation by capturing just one image. The feedback of the controller is totally based on vision and is obtained through digital image processing techniques. The algorithms used to process the omnidirectional images improve the system robustness and performance.

Simulations were carried out in order to study the controller behaviour facing different group formations. The results obtained through the experiments with real robots show that the proposed control system has achieved a good performance.

1 Introdução

O crescimento tecnológico dos últimos anos tem possibilitado grandes avanços no campo da robótica. Ano após ano, é disponibilizada no mercado uma variedade de processadores cada vez mais rápidos, menores, econômicos e com mais memória. Além disso, a tecnologia de materiais exerce um importante papel, possibilitando a construção de peças mais leves e resistentes. Esse avanço da robótica se deve principalmente aos resultados obtidos pelos pesquisadores nas últimas décadas, pois o estudo da robótica não está mais associado apenas aos aspectos computacionais e construtivos, mas abrange várias outras áreas do conhecimento, como teoria de controle (linear e não linear), identificação de sistemas, modelamento cinemático e dinâmico, engenharia de *software*, biologia (anatomia e comportamento social dos animais), ciências sociais (teoria da organização, psicologia cognitiva e até mesmo economia), entre outras [1, 3].

Apesar da tendência da maioria das ciências em se subdividir à medida em que avançam, a robótica continua a ser dividida em robótica de manipulação, ou industrial, e robótica móvel. No primeiro caso, o maior número de aplicações está nas indústrias, onde os robôs executam trabalhos como montagem, pintura e soldagem. Poucos são equipados com algum tipo de sensor que permite uma percepção detalhada do ambiente para a tomada de decisões, como visão computacional. Já o campo da robótica móvel é a área que vem apresentando os maiores avanços e expandindo sua área de atuação. Um exemplo disso são os robôs exploradores Spirit e Opportunity [4], construídos pela NASA, que recolhendo amostras do solo marciano em lados opostos do planeta confirmaram que um dia houve água na superfície de Marte. A Figura 1.1 - (a) mostra um desses robôs, o Spirit. Outros exemplos podem ser citados, como o AIBO (*Artificial Intelligence roBOt*), da Sony, e o RMAX, da Yamaha, também mostrados na Figura 1.1. O primeiro, embora projetado para ser um robô de estimação, também foi usado em competições de futebol de robôs; o segundo é um helicóptero capaz voar horizontalmente segundo uma trajetória pré-determinada, mas que pode ser modificada durante o vôo. No Japão, foi utilizado para observar as redondezas de vulcões através de câmaras e sensores embarcados para deteção de gases [1].



(a) (b)

(c)

Figura 1.1: (a) Robô-jipe Spirit, que iniciou sua exploração em Marte no início de 2004; (b) AIBO, capaz de aprender com o dono e com o ambiente, além de desenvolver personalidade ao longo do tempo. Infelizmente, a Sony encerrou a sua produção no início de 2006; e (c) RMAX, com 3,63 m de comprimento e 1,08 m de altura, possui autonomia de vôo de uma hora transportando uma carga de até 30 kg [1].

Entretanto, por mais avançado que seja um determinado robô, existem tarefas nas quais é conveniente a utilização de mais de um robô, como mapeamento, o combate a incêndios, o resgate de sobreviventes após um acidente ou desastre natural, a vigilância de instalações industriais ou militares, monitoramento (e vigilância) de regiões de fronteiras para inibir a entrada ilegal de imigrantes, o contrabando e o tráfico de drogas, a limpeza de grandes áreas como, por exemplo, campos minados abandonados após uma guerra, o transporte de objetos grandes e pesados. Além disso, um robô equipado com toda a capacidade de processamento e sensores necessários para realizar, sozinho, diferentes tarefas do mundo real, apresenta um custo muito elevado.

Tudo isso motivou, a partir da década de 1980, os estudos dos chamados Sistemas Multi-robôs. Até então, os esforços eram concentrados em pesquisas de sistemas distribuídos para solução de problemas [5] ou formados por apenas um robô. Os primeiros deram origem ao que hoje é conhecido como Sistemas Multi-agentes. Em [6], o autor afirma que um agente deve ser capaz de perceber sozinho tanto do que precisa quanto de realizar as ações necessárias para atingir os seus objetivos, sem a necessidade de ser instruído a todo momento. Além disso, um

sistema multi-agente é formado por um certo número de agentes que interagem uns com os outros, normalmente através de uma rede computacional de apoio.

Sem dúvida, essa idéia está intimamente relacionada com sistemas multi-robôs. Entretanto, a teoria de sistemas multi-agentes está mais focada em agentes puramente computacionais. Em outras palavras, robôs reais são capazes de interagir com o ambiente no qual estão inseridos através da aplicação de forças, da recepção de sinais, como na visão, ou mesmo através da emissão de um sinal e recepção do respectivo eco, como no caso dos sensores de ultra-som e *laser*. Agentes computacionais, porém, não possuem essas propriedades. Em sistemas multi-robôs, por exemplo, o controle de formação é essencial para um bom desempenho da equipe, preocupação que não existe em sistemas cujos agentes são puramente virtuais. Mesmo assim, fica claro que as teorias de sistemas multi-agentes e multi-robôs se sobrepõem em várias questões, permitindo que uma aproveite os avanços alcançados pela outra.

Dentre os primeiros esforços nas pesquisas em sistemas multi-robôs, podem ser citados os trabalhos realizados em [7, 8, 9, 10, 11, 12]. Desde então, muitos avanços foram conquistados nesta área, abordando novos temas, expandindo e criando novos conceitos. Em [5], os autores apontam boas fontes de informação, como as coleções [13, 14], os anais [15, 16, 17], além das publicações [18] sobre colônias de robôs e [19] sobre sistemas multi-robôs heterogêneos. Todas essas fontes, juntas, fornecem um bom panorama das pesquisas mais recentes na área bem como algumas tendências.

Atualmente, as pesquisas em sistemas multi-robôs podem ser divididas nos seguintes ramos:

1. Inspirações biológicas
2. Comunicação
3. Arquiteturas, alocação de tarefas e controle
4. Localização, mapeamento e exploração
5. Manipulação e transporte de objetos
6. Coordenação de movimento
7. Robôs reconfiguráveis
8. Aprendizado

Em [5] e [20], os principais trabalhos relativos a cada um destes tópicos são apresentados e discutidos. Chama-se a atenção, aqui, para três tópicos: inspirações biológicas, aprendizado e coordenação de movimento. O primeiro merece destaque por ter impulsionado grande parte dos trabalhos em robótica móvel cooperativa até hoje realizados através do conceito de controle baseado em comportamento [21, 22]. O segundo tópico, por ser uma área que tem muito a contribuir e que ainda não foi bem explorada. Trabalhos abordando o aprendizado por imitação [23] e exploração em busca de recursos [24] podem ser citados.

O terceiro ponto de destaque é coordenação de movimento, tema abordado em [25], que utiliza estruturas virtuais para obter boa precisão na formação, e [26], cujo *framework* desenvolvido permite a realização de diversas tarefas em cooperação e a estabilidade da abordagem é provada. Além disso, [27] descreve como a teoria de controle descentralizado pode ser usada para analisar o desempenho do controle da formação de múltiplos robôs cooperativos; [28] propõe um cenário onde os robôs entram e mantêm uma formação sem a necessidade de comunicação; [29] apresenta uma abordagem probabilística para a coordenação de múltiplos robôs em tarefas de exploração e [30] aborda o problema conhecido como CMOMMT (*Cooperative Multirobot Observation of Multiple Moving Targets*) em ambientes como escritório com vários corredores, usando robôs móveis em conjunto com sensores fixos. Outros trabalhos interessantes podem ser encontrados em [31, 32, 33, 34, 35, 36, 37]. Todos os estudos citados relativos a este último tópico deixam claro que coordenar o movimento e, mais especificamente, controlar a formação, é fundamental para a realização de tarefas em conjunto. Alguns deles serão abordados em mais detalhes ao longo deste texto. Outros assuntos pertinentes e interessantes são encontrados em [3], como as vantagens de um robô ser capaz de modelar outros membros do grupo e uma discussão a respeito de como o comportamento cooperativo surge (ou deve surgir) sem a intervenção humana, como ocorre em sociedades de insetos, onde tal comportamento é vital para a sobrevivência da colônia.

1.1 Motivação

Como mencionado anteriormente, a limitação espacial de um único robô e a necessidade da realização de tarefas cada vez mais complexas motivaram o estudo dos sistemas multi-robôs. A idéia é que um time de robôs mais simples possa realizar uma tarefa, através de redundância e cooperação, de forma mais confiável, rápida ou barata do que seria possível com apenas um [5]. De fato, além das tarefas já mencionadas, esses sistemas tendem a apresentar desempenho superior àquele demonstrado por um único robô em outras tarefas do mundo real, como [1]:

- Exploração em ambientes perigosos para os seres humanos. Por exemplo, no estudo do interior de crateras de vulcões em atividade e das formas de vida existentes em altas profundidades marinhas, no mapeamento e análise da superfície de outros planetas, na manutenção em satélites artificiais em órbita, entre outros.

- Criação de uma rede de comunicação dinâmica com o objetivo é fornecer infraestrutura para cobrir uma determinada área da melhor maneira possível, onde os nós da rede seriam robôs especializados em comunicação, que mudariam sua configuração espacial automaticamente, adaptando a rede às variações do ambiente e reduzindo a probabilidade de interrupção no tráfego de informação.

- Rede de sensores altamente distribuída, onde grandes colônias de robôs simples e baratos cobririam grandes áreas para o monitoramento, por exemplo, da temperatura da água nos oceanos ou de abalos sísmicos ao redor da Terra, o que permitiria prever a formação de *tsunamis* e a conseqüente evacuação das regiões a serem atingidas.

- Construção/montagem de bases científicas na superfície ou no subsolo de planetas e satélites naturais do sistema solar para o estudo destes astros ou de outras galáxias.

Existem, ainda, as competições entre times de robôs, como RoboCup e MIROSOT, cujo objetivo maior é promover o avanço da robótica móvel. Os resultados dessas competições e as publicações que delas surgem têm mostrado que o objetivo vem sendo alcançado.

Na grande maioria – senão na totalidade – dos exemplos de cooperação citados, o controle da formação do grupo é fundamental para que a tarefa seja cumprida de forma eficiente. No final da seção anterior foram citados vários trabalhos já realizados nesta área que mostram a importância do controle de formação para a cooperação robótica. É fácil encontrar também na atividade humana ou mesmo na natureza exemplos da importância da manutenção de uma formação desejada. Tarefas como busca e salvamento, transporte de cargas, vigilância de perímetros, entre outras, quando realizadas por humanos, requerem uma formação adequada. Pássaros migratórios mantêm uma formação em *V* porque isso reduz o gasto de energia durante o vôo. Em 2001 [38], foi demonstrado que pelicanos brancos treinados para voar em formação tiveram seus batimentos cardíacos (e, portanto, a energia dispendida) reduzidos em cerca de 30% [1].

Por outro lado, controlar a formação de grupos de robôs é mais difícil do que controlar apenas um. Em um grupo de robôs móveis, cada membro pode interferir ou operar dentro da área de atuação ou raio de alcance de um outro membro do grupo, tornando-se, uns para os outros, obstáculos móveis. Com isso, é mais difícil resolver o problema do desvio de obstáculos.

Outro problema típico de sistemas multi-robôs é como a equipe deve agir diante da falha de um de seus integrantes de modo a garantir que a tarefa seja concluída com sucesso. Em muitos casos, a falha de um dos robôs compromete seriamente ou mesmo impede a conclusão da tarefa, como é o caso de dois robôs transportando uma carga grande e pesada. Tarefas deste tipo são conhecidas como Tarefas Fortemente Acopladas (*Tightly Coupled Tasks*) [39, 40].

Um ponto importante e que deve ser considerado em sistemas multi-robôs é a escalabilidade da abordagem adotada, ou seja, é necessário discutir se uma mesma abordagem pode ser aplicada em grupos com dezenas, centenas ou milhares de robôs. Muitas abordagens se mostram atrativas para equipes com poucos integrantes, mas não são adequadas para grupos com, por exemplo, mais de dez robôs.

Como pode ser visto, sistemas multi-robôs apresentam muitas vantagens sobre aqueles que utilizam apenas um. Entretanto, alguns inconvenientes podem ser observados, como o aumento na complexidade do controle e, dependendo do tipo de tarefa, a dependência de outros membros do grupo. Claramente, há muito para ser feito no campo de sistemas multi-robôs.

Assim como ocorre com os animais, os robôs podem ser dotados de vários tipos de sensores. O tipo, a sensibilidade, a precisão e a quantidade depende de sua utilidade: servem tanto para medir o estado interno – como pressão sangüínea ou temperatura, por exmplo – como para obter informações do meio externo. Os sensores desempenham papel fundamental tanto na natureza quanto em robótica. Em ambos os casos, são usados ainda para fechar malhas de controle de posição, velocidade, aceleração, temperatura, pressão, etc.

Vários sensores construídos pelo homem tentam imitar aqueles existentes na natureza, como é o caso das câmaras de vídeo, que aproximam o sentido da visão, um dos mais comuns e importantes. O seu sucesso se deve à grande quantidade de informação que é capaz de fornecer. Alguns animais utilizam a visão para navegar mantendo uma determinada formação, como é o caso das aves migratórias e de algumas espécies de peixes, que, nadando próximos uns dos outros, fazem o cardume parecer um único e grande animal, confundindo e intimidando o predador. Em sistemas multi-robôs, a visão computacional também representa uma boa maneira de navegar mantendo uma formação desejada. Para um bom controle de formação é necessário ter uma boa estimativa da postura (posição e orientação) dos outros membros do grupo. Além disso, sistemas de visão que fornecem um amplo campo visual aumentam a percepção do ambiente, de outros robôs e objetos, tornando-se atrativos para o controle de formação, cooperação e execução de tarefas em robótica.

Uma maneira de ampliar o campo de visão é através do uso de imagens omnidirecionais (360^o de campo visual horizontal) [41]. Mais uma vez, a inspiração veio da natureza. Várias

espécies de insetos são dotadas de visão omnidirecional. Um exemplo é mostrado na Figura 1.2, onde é possível observar que o sistema de visão maximiza o campo visual.

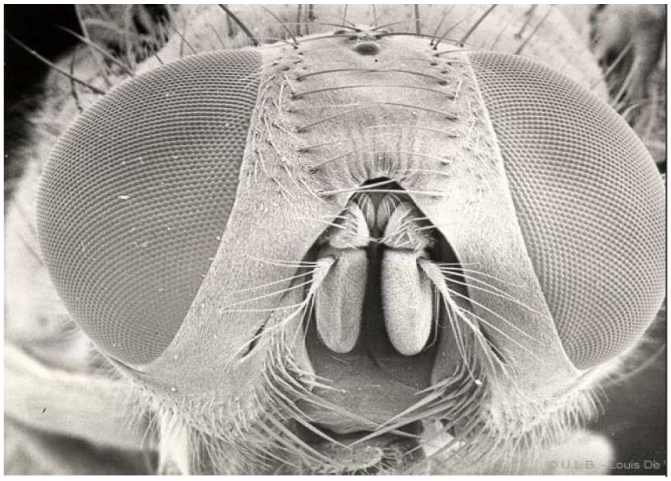


Figura 1.2: Evolução: sistema de visão maximiza campo visual de várias espécies de insetos [2].

Imagens omnidirecionais podem ser obtidas através de sistemas catadióptricos, formados a partir do acoplamento de espelhos (hiperbólicos, elípticos ou parabólicos) e lentes (câmaras de vídeo) [42]. A Figura 1.3 mostra um exemplo de um sistema de visão omnidirecional construído pelo homem.

Em sistemas multi-robôs, imagens omnidirecionais são usadas de duas maneiras: na primeira delas, todos os membros da equipe possuem seu próprio sistema omnidirecional. Na segunda, apenas um robô, geralmente denominado líder da equipe, transporta um sistema desse tipo. Neste caso, a estrutura de controle adotada é, normalmente, do tipo centralizado. A próxima seção traz informações mais detalhadas sobre os tipos de estruturas de controle adotadas em sistemas multi-robôs.

1.2 Controle Centralizado *x* Controle Descentralizado

O controle de um grupo de robôs pode ser feito, basicamente, de três maneiras: controle centralizado, descentralizado ou controle misto [1].

- Controle centralizado

  É o controle hierárquico onde um líder é o responsável por tomar as decisões a fim de atingir o objetivo. Isso não quer dizer que deve haver apenas um líder. Assim como acontece dentro de empresas e nas forças armadas, um membro pode ser responsável por

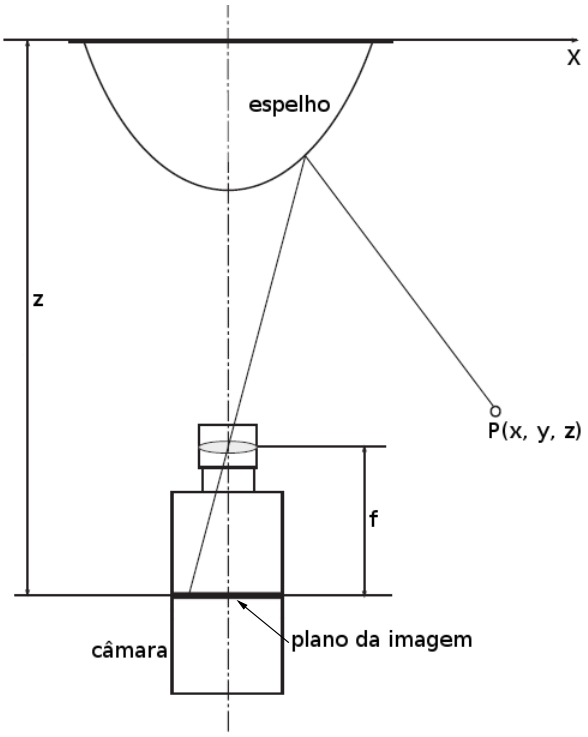


Figura 1.3: Exemplo de sistema de visão omnidirecional. $P(x,y,z)$ representa um ponto no mundo, f é a distância focal da câmara e z é a distância entre o foco do espelho e o plano da imagem. *Adaptado de* [2].

um grupo, estabelecendo diferentes níveis de controle. Normalmente, quanto mais alta for sua hierarquia, maior deve ser sua capacidade de processamento e conhecimento dos objetivos a serem alcançados. Esta é uma opção eficiente e barata para grupos com um pequeno número de integrantes [1]. Entretanto, uma falha do líder pode representar a falha de todo o sistema. Neste ponto, vale ressaltar a importância do aprendizado em robótica: no caso de uma falha do líder, algum membro do grupo, já tendo observado e aprendido suas principais funções, pode assumir o seu lugar, permitindo a continuidade do trabalho. Alguns exemplos de sistemas que utilizam controle centralizado são encontrados em [33, 34, 35, 36, 37].

- Controle descentralizado

  Quando se trata de um grupo com um grande número de robôs, normalmente é adotado um controle do tipo distribuído. Entretanto, nada impede que seja usado em grupos com poucos integrantes. Este tipo de controle foi inspirado nas colônias de insetos, onde cada membro opera usando apenas informação local, através de regras baseadas em comportamentos. Este tipo de controle se mostra mais robusto que o anterior, mas, geralmente, exige que todos os robôs possuam sensores e capacidade de processamento suficientes, aumentando o custo da equipe. Ultimamente, os pesquisadores têm mostrado maior inte-

resse neste tipo de controle, com vistas a um futuro onde grandes grupos de robôs realizarão tarefas complexas. Os trabalhos [26, 27, 28, 31] são bons exemplos de aplicações que utilizam o controle descentralizado.

- Controle Misto

  Menos comum na literatura, o controle misto é uma estratégia bastante interessante [43]: grandes grupos de robôs podem ser divididos em equipes menores que, internamente, são controladas por um líder (controle centralizado). A cooperação entre elas e a coordenação das tarefas são feitas pelos líderes de cada equipe, implementando, assim, um controle descentralizado, configurando um controle misto. Espera-se que, nos próximos anos, arquiteturas deste tipo se tornem mais comuns.

1.3 Cooperação

Muitas vezes, para que tarefas sejam cumpridas por uma equipe de robôs é necessário haver cooperação. De acordo com [20], o campo da robótica móvel cooperativa é tão novo que nenhum tópico desse assunto pode ser considerado suficientemente maduro. Prova disso é que ainda não existe uma definição formal para "cooperação robótica". É possível encontrar, na literatura, diferentes definições para cooperação, como [3, 44]:

- comportamento colaborativo conjunto direcionado a um objetivo no qual existe um interesse comum ou recompensa [45];
- uma forma de interação, geralmente baseada em comunicação [24];
- trabalho em conjunto para fazer algo que cria um resultado progressivo, como aumento no desempenho ou ganho de tempo [9];

A terceira definição, além de mais intuitiva, deixa clara uma importante vantagem da maioria dos sistemas cooperativos: o aumento no desempenho. A partir deste ponto, o termo "cooperação robótica", ou apenas "cooperação", será usado de acordo com essa definição. Mesmo não havendo, ainda, uma definição formal, os avanços na teoria da organização, na psicologia cognitiva e no ramo da biologia que estuda o comportamento social dos animais podem fornecer indícios para a formalização do conceito de cooperação robótica.

1.4 Objetivo

O objetivo desta Dissertação de Mestrado é o controle de formação de uma equipe de robôs móveis de baixo custo para a realização de tarefas em cooperação. É proposta uma estratégia de controle centralizado baseada no processamento de imagens omnidirecionais e em técnicas de controle não linear.

A equipe é composta por um líder dotado de maior capacidade de processamento que transporta o sistema de visão omnidirecional, enquanto os outros componentes são robôs mais simples e baratos, que possuem apenas um microcontrolador e os dispositivos necessários para o acionamento dos motores. Essa restrição no custo é a responsável pela adoção do controle centralizado, uma vez que somente o líder é capaz de processar as informações necessárias para o controle de formação.

O uso da teoria de controle não linear permite o projeto de um controlador de postura robusto e estável, que garante a convergência dos robôs para a formação desejada. Essa teoria permite, ainda, que a estabilidade do controlador seja matematicamente provada através do método de Lyapunov.

O controlador utilizado neste trabalho possui realimentação puramente visual. Assim, um amplo campo de visão é fundamental para um bom desempenho, daí a opção por um sistema de visão omnidirecional. O fato deste sistema de visão estar embarcado no líder e mover-se junto com a equipe torna a área de trabalho ilimitada, uma importante característica desta abordagem.

Algumas considerações a respeito desta equipe precisam ser feitas. Com exceção do líder, os outros robôs são conhecidos como robôs celulares, dada a sua simplicidade e tamanho. É importante notar que esta nomenclatura não tem relação direta com um *Sistema Robótico Celular* [8], onde cada robô é responsável por uma área de atuação ou célula de trabalho. Além disso, esta equipe é, por natureza, heterogênea, já que o líder é diferente dos robôs celulares em vários aspectos. Na verdade, os próprios robôs celulares são, entre si, heterogêneos, pois suas características são próximas, mas não idênticas.

1.5 Metodologia

Inicialmente, o único robô disponível era o líder. Para completar a equipe, dois robôs celulares foram construídos com recursos do Laboratório de Automação Inteligente (LAI), pertencente ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Espírito Santo (UFES). Paralelamente à construção dos robôs celulares, o controlador foi imple-

mentado usando a linguagem de programação C++ para que as primeiras simulações pudessem ser feitas.

A determinação das posturas dos robôs celulares conta com a utilização de imagens omnidirecionais, que permitem a visualização de toda a equipe capturando apenas uma imagem, juntamente com técnicas de processamento digital de imagens. A posição é estimada com um algoritmo de segmentação de cores, que identifica os robôs celulares por meio de cartões coloridos colocados sobre eles. Já a orientação é estimada por um algoritmo que se baseia na trajetória descrita.

As posturas dos robôs celulares, assim que determinadas pelo processamento das imagens omnidirecionais, são repassadas a um controlador não linear estável, que gera novos sinais de comando, conduzindo a equipe à formação desejada. A estabilidade desse controlador foi provada com o método de Lyapunov.

Várias simulações foram realizadas para verificar o comportamento do controlador e fazer um primeiro ajuste dos seus ganhos. Os experimentos feitos em seguida, onde diferentes formações foram impostas, confirmaram a validade deste trabalho.

1.6 Estrutura da Dissertação

Este trabalho está organizado em cinco capítulos. Segue-se uma breve descrição do que é apresentado e discutido em cada um desses capítulos.

- Capítulo 1: Introdução

  É o presente capítulo, que traz, no início, um panorama com algumas das pesquisas consideradas mais relevantes em sistemas multi-robôs. Em seguida, é apresentada a motivação deste trabalho, bem como de vários outros relativos ao mesmo assunto. Após uma discussão sobre as principais estruturas de controle usadas e algumas definições existentes para cooperação robótica, o objetivo desta Dissertação é apresentado.

- Capítulo 2: O Controlador

  Este capítulo é dedicado ao controlador não linear utilizado neste trabalho. Após a motivação, são derivadas as leis de controle que o compõem. Em seguida, essas leis são discutidas em detalhes através da análise de suas equações e de diferentes situações ou configurações nas quais a equipe pode se encontrar. Por fim, é apresentada a prova de estabilidade deste controlador.

- Capítulo 3: O Processamento de Imagens

  O objetivo deste capítulo é apresentar as várias técnicas de processamento de imagens que foram empregadas, bem como a filtragem e o refinamento de dados empregados a fim de melhorar as estimativas de postura de cada robô seguidor. Também são abordados e discutidos os algoritmos desenvolvidos para complementar as técnicas utilizadas.

- Capítulo 4: Resultados Experimentais

  Neste capítulo, os resultados das simulações e dos experimentos com uma equipe de robôs reais são apresentados e analisados, comprovando a estabilidade e bom desempenho do controlador empregado.

- Capítulo 5: Conclusões

  Este capítulo encerra esta Dissertação com uma discussão geral do trabalho realizado, as principais contribuições, as dificuldades encontradas e sugestões para os trabalhos futuros.

2 O Controlador

2.1 Introdução

Para que uma equipe de robôs móveis seja capaz de navegar em um ambiente mantendo uma determinada formação geométrica de forma estável é necessária a existência de um controle. Este, como discutido no capítulo anterior, pode ser do tipo centralizado ou descentralizado. Neste trabalho foi utilizado um controle centralizado, uma vez que uma das propostas é a construção de uma equipe de robôs de baixo custo, o que limita, além da qualidade e quantidade de sensores, a capacidade computacional que se pode embarcar em cada componente do grupo. Portanto, a maioria dos robôs não possui os sensores e a capacidade de processamento necessários para a implementação de uma arquitetura de controle descentralizado.

Desta forma, apenas um membro da equipe é dotado de capacidade de processamento suficiente para a implementação do controlador. Este é designado como o líder e será responsável pela navegação e controle de formação do grupo. O restante da equipe é formado por robôs celulares, chamados aqui de seguidores, pois apenas executam os comandos enviados pelo líder.

Para que o líder possa controlar a formação da equipe, ele precisa conhecer o estado de cada um dos outros robôs. Neste trabalho, o estado de um seguidor é definido como sua postura relativa ao líder, ou seja, sua posição e sua orientação relativas ao robô líder. A Figura 2.1 ilustra a representação da postura de dois seguidores no sistema de coordenadas do líder.

De acordo com a Figura 2.1, a posição do i-ésimo seguidor pode ser representada como na Equação 2.1.

$$\xi_i = \begin{pmatrix} x_i \\ y_i \end{pmatrix} = \begin{pmatrix} x_i = r_i cos(\gamma_i) \\ y_i = r_i sen(\gamma_i) \end{pmatrix} \tag{2.1}$$

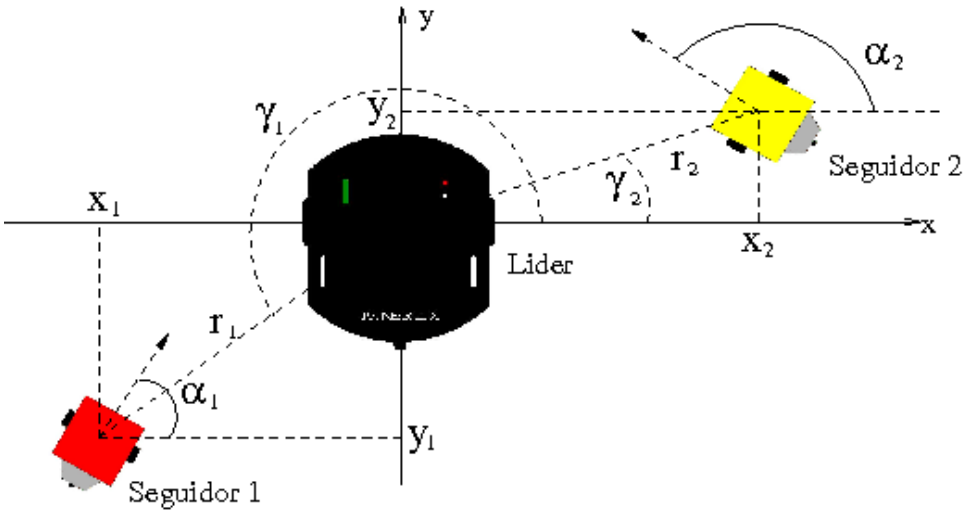


Figura 2.1: Representação da postura de dois seguidores no sistema de coordenadas do líder.

Assim, a postura do i-ésimo seguidor é dada pelo vetor χ_i, de acordo com a Equação 2.2.

$$\chi_i = \begin{pmatrix} x_i \\ y_i \\ \alpha_i \end{pmatrix} = \begin{pmatrix} r_i cos(\gamma_i) \\ r_i sen(\gamma_i) \\ \alpha_i \end{pmatrix} \tag{2.2}$$

Controlar a formação de uma equipe de robôs significa, em outras palavras, assegurar que cada um possua uma determinada postura em relação aos outros integrantes da equipe. Portanto, o objetivo principal deste trabalho é controlar a postura de cada seguidor relativa ao líder.

De acordo com a teoria de controle [46], para um sistema descrito por equações não lineares, a adoção de um controlador linear requer a linearização deste sistema nas vizinhanças do ponto de operação. Dependendo da não linearidade da região onde este ponto se encontra, a estabilidade do controlador fica limitada a uma vizinhança muito restrita, comprometendo a eficiência do controle adotado. Em outras palavras, o desempenho do controlador pode se tornar extremamente dependente das condições iniciais do sistema e o controlador pode apresentar alta sensibilidade a ruídos, pois, se uma perturbação qualquer retirar o sistema desta vizinhança onde o controle é estável, não há garantia de que o controlador conduzirá o sistema novamente ao ponto de operação. Além disso, determinar com precisão qual é esta vizinhança não é, em geral, uma tarefa fácil.

Claramente, como pode ser visto na Equação 2.2, a postura dos seguidores é descrita por equações não lineares. Isso motivou a adoção de um controlador não linear, uma vez que se deseja controlar a formação da equipe independentemente de sua forma geométrica, dos erros iniciais de posição e orientação e do ruído presente nas estimativas de postura dos seguidores.

Para controlar as posturas dos seguidores é necessário estimá-las, ou seja, realimentar o controlador. O único sensor do qual o robô líder dispõe para estimar as posturas dos seguidores é o sistema de visão omnidirecional montado sobre sua plataforma, exibido na Figura 2.2. Portanto, trata-se de um controle não linear com realimentação puramente visual.

Figura 2.2: Sistema omnidirecional responsável pela realimentação do controlador.

Assim, o controlador proposto baseia-se nas estimativas da postura de cada seguidor, obtidas através do processamento das imagens omnidirecionais, para gerar os respectivos sinais de controle. Estes sinais nada mais são do que as velocidades linear e angular que cada seguidor deve ter para entrar e permanecer na formação desejada.

O controlador utilizado neste trabalho pode ser subdividido em dois controladores menores. O primeiro deles, chamado Controlador de Formação, considera apenas o erro de formação de um seguidor enquanto o segundo, chamado Controlador de Compensação, incorpora o movimento do robô líder na determinação dos sinais de controle.

2.2 O Controlador de Formação

O processamento das imagens omnidirecionais capturadas pelo sistema de visão fornece uma estimativa da posição de cada seguidor no referencial do líder, como na Equação 2.1. Um

único vetor que contenha todas as coordenadas das posições dos seguidores pode ser escrito como na Equação 2.3.

$$\xi = \begin{pmatrix} \xi_1 \\ \xi_2 \\ \vdots \\ \xi_n \end{pmatrix} \tag{2.3}$$

Embora a estimativa de posição seja conseguida com o auxílio das imagens omnidirecionais, o desempenho do controlador deve ser independente do sensor utilizado para obter esta informação. De acordo com a aplicação em questão, pode ser necessário algum tipo de transformação nas coordenadas obtidas pelo sensor usado. Por exemplo, uma transformação pode ser usada para corrigir as disparidades de um sistema de visão perspectiva ou para trabalhar com parâmetros de formação diferentes das posições mas associados à formação desejada, como a distância entre os robôs da equipe, o baricentro da formação, etc. Além disso, se o controlador for dependente de medições feitas diretamente na imagem, caso o sistema de visão sofra alguma modificação ou seja trocado por outro, os parâmetros do controlador teriam que ser sintonizados novamente. Assim, para abordar o problema de uma forma genérica, pode-se considerar o vetor de *parâmetros de formação* como $\rho(\xi)$. Daí o cuidado tomado em chamar este controlador de Controlador de *Formação*, e não de *Posição*.

Neste trabalho, para tornar o controlador independente das medições e características do sistema de visão, todas as posições dos seguidores fornecidas pelas imagens omnidirecionais são convertidas para posições no mundo real, de acordo com a Figura 2.1. Em outras palavras, uma transformação $\Gamma : Z^2 \rightarrow \Re^2$, definida através de um conjunto de equações polinomiais, converte as posições (dadas em *pixels*) do plano da imagem para posições (dadas em metros) no plano de trabalho da equipe. O resultado desta transformação é o vetor ξ, ou seja, o vetor que representa as coordenadas dos seguidores no mundo real e relativas ao líder. É importante ressaltar que Γ não faz parte do grupo de transformações responsáveis pela adoção da notação $\rho(\xi)$, já que estas últimas são aplicadas ao vetor ξ, enquanto Γ fornece ξ. O motivo para esta distinção está no fato de que o controlador foi definido no referencial do líder, ou seja, no mundo, e não na imagem. Caso fosse definido no referencial da imagem, o controlador seria do tipo servo-visual.

Uma conseqüência importante da definição do controlador no referencial do líder e de usar diretamente as posições dos robôs é que não é necessário realizar nenhuma transformação nas coordenadas dos seguidores para controlar a formação, neste caso, $\rho(\xi) = \xi$. Contudo, a notação $\rho(\xi)$ será mantida por questões de generalidade.

Sendo $\rho(\xi)$ um vetor de parâmetros de formação, o vetor de velocidades de formação é obtido derivando-se $\rho(\xi)$ em relação ao tempo, como na Equação 2.4.

$$\dot{\rho} = J(\xi)\dot{\xi} \tag{2.4}$$

onde $J(\xi)$ é o jacobiano de ξ.

A partir da Equação 2.4 pode-se projetar um controlador para que a equipe entre em formação. A lei de formação é dada pela Equação 2.5.

$$\dot{\xi}_{fr} = J^{-1}(\xi)\left(\dot{\rho}_d + f_{\tilde{\rho}}(\tilde{\rho})\right) \qquad com \qquad \tilde{\rho} = \rho_d - \rho \tag{2.5}$$

onde $\tilde{\rho}$ é o vetor de erros de formação [36], ρ_d é o vetor com os parâmetros de formação desejados e ρ o vetor com os valores atuais dos parâmetros de formação. O vetor $\dot{\rho}_d$ representa a taxa de variação temporal dos parâmetros desejados e será não nulo quando os parâmetros desejados de pelo menos um seguidor estiverem variando no tempo. Caso contrário, $\dot{\rho}_d$ será sempre um vetor nulo.

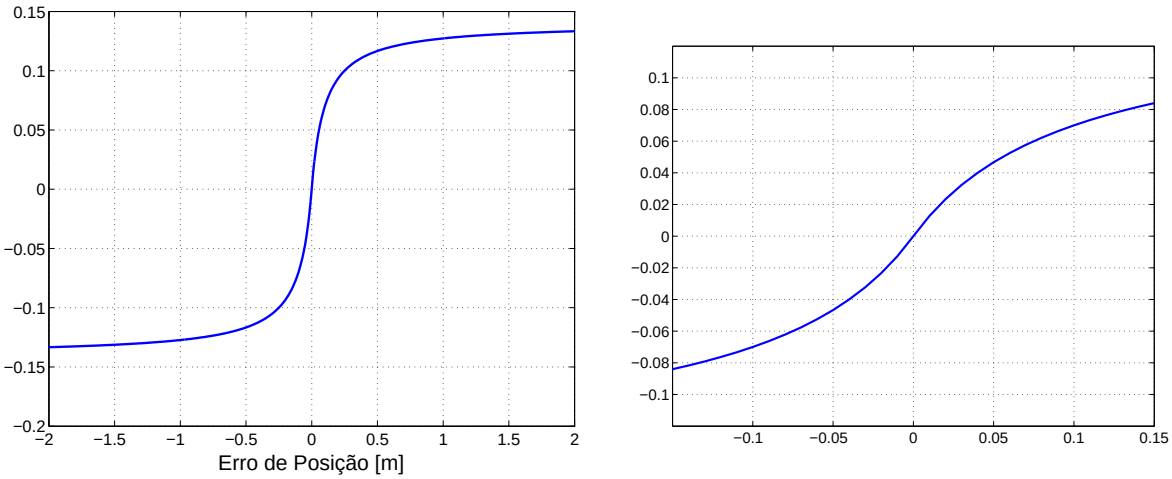
O vetor $\dot{\xi}_{fr}$ representa as velocidades de formação de referência, ou seja, as velocidades no sistema de coordenadas do líder que os seguidores devem ter para alcançar a formação. Nota-se, ainda na Equação 2.5, que para obter $\dot{\xi}_{fr}$ é necessário determinar primeiro o jacobiano inverso $J^{-1}(\xi)$. Como se sabe, o cálculo de inversão matricial é, em geral, computacionalmente custoso. Entretanto, como neste trabalho nenhuma transformação precisa ser aplicada a ξ, $J(\xi)$ e $J^{-1}(\xi)$ são matrizes identidade. Logo, o custo computacional da determinação de $\dot{\xi}_{fr}$ não é significativo.

A função $f_{\tilde{\rho}}(\tilde{\rho})$ é uma função de saturação sobre o erro de formação. Em [35], $f_{\tilde{\rho}}(\tilde{\rho})$ é dada pela Equação 2.6.

$$f_{\tilde{\rho}}(\tilde{\rho}) = diag\left[\frac{k_f}{a + |\tilde{\rho}_j|}\right]\tilde{\rho} \tag{2.6}$$

onde k_f representa o valor de saturação e a é tal que $\frac{k_f}{a}$ representa o ganho para erros pequenos. A forma de $f_{\tilde{\rho}}(\tilde{\rho})$ assim definida pode ser vista na Figura 2.3. Vale ressaltar que, na Figura 2.3 - (a), as escalas para os eixos x e y são diferentes. A Figura 2.3 - (b) exibe, na mesma escala, o comportamento de $f_{\tilde{\rho}}(\tilde{\rho})$ para erros de formação menores do que 15 cm, mostrando a verdadeira inclinação desta função para estes erros.

O objetivo de se usar $f_{\tilde{\rho}}(\tilde{\rho})$ é evitar que erros grandes de formação causem velocidades de formação também elevadas, submetendo os motores a tensões acima dos valores nominais ou mesmo causando um mal funcionamento do robô.



(a) (b)

Figura 2.3: (a) Formato da função $f_{\tilde{\rho}}(\tilde{\rho})$ para $k_f = 0.14$ e $a = 0.1$ e (b) *Zoom* para observar o comportamento de $f_{\tilde{\rho}}(\tilde{\rho})$ para erros de formação menores que 15 cm.

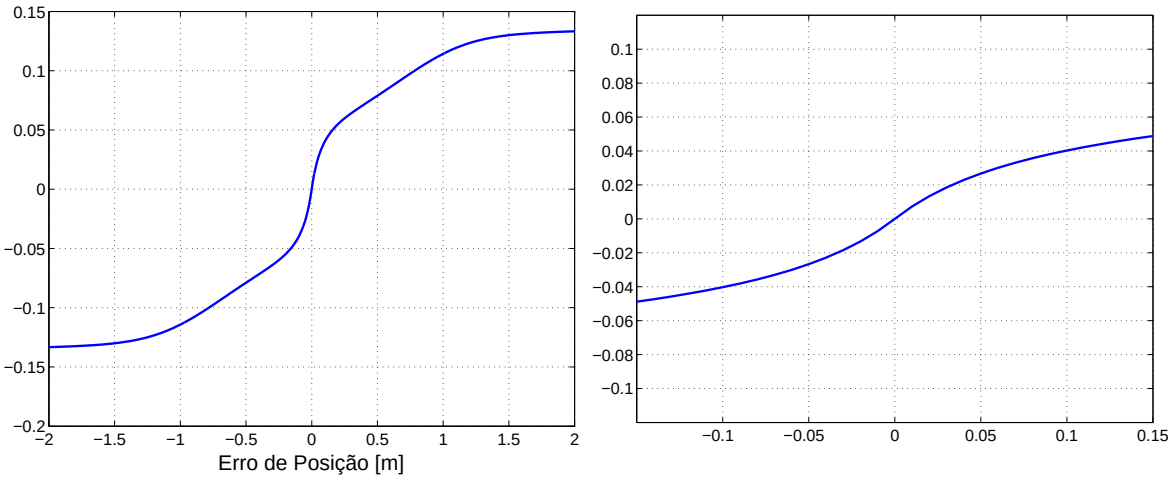
Entretanto, o ganho $\frac{k_f}{a}$ provoca oscilações na velocidade linear dos seguidores quando estes estão próximos das posições desejadas. Esse fato, acompanhado de um erro angular – o que é freqüente – pode fazer com que o seguidor se afaste da posição desejada, causando oscilações também na trajetória. Isso ocorre porque $\frac{k_f}{a}$ é grande para erros de formação pequenos (ver Figura 2.3). Reduzir o valor de k_f implica em limitar a velocidade dos seguidores em um valor mais baixo, o que não é interessante. Uma solução intuitiva seria aumentar o valor de a. Como resultado, os seguidores podem não alcançar as posições desejadas quando toda a equipe estiver navegando, uma vez que o ganho do controlador de formação foi reduzido.

Assim, neste trabalho, propõe-se uma forma ligeiramente diferente para a função $f_{\tilde{\rho}}(\tilde{\rho})$, mostrada na Equação 2.7.

$$f_{\tilde{\rho}}(\tilde{\rho}) = diag\left[\frac{k_f(\tilde{\rho}_j)}{a+\|\tilde{\rho}_j\|}\right]\tilde{\rho} \quad \text{com} \tag{2.7}$$

$$k_f(\tilde{\rho}_j) = k_{f1} + k_{f2} tanh\left(\|\tilde{\rho}_j\|\right).$$

A Equação 2.7 permite maior controle sobre os parâmetros que definem o comportamento de $f_{\tilde{\rho}}(\tilde{\rho})$. Agora, $k_{f1}+k_{f2}$ representa o valor de saturação e o ganho para erros pequenos é dado por $\frac{k_{f1}}{a}$. Desta forma, é possível manter o valor de saturação e o parâmetro a inalterados e, ao mesmo tempo, obter um ganho pequeno para erros de formação pequenos, eliminando as oscilações na velocidade linear dos seguidores. A forma de $f_{\tilde{\rho}}(\tilde{\rho})$ definida segundo a Equação 2.7 encontra-se na Figura 2.4.



(a) (b)

Figura 2.4: (a) Formato da função $f_{\tilde{\rho}}(\tilde{\rho})$ para $k_{f1} = 0.08$, $k_{f2} = 0.06$ e $a = 0.1$ e (b) *Zoom* para observar o comportamento de $f_{\tilde{\rho}}(\tilde{\rho})$ para erros de posição menores que 15 cm.

Comparando as Figuras 2.3-(b) e 2.4-(b), nota-se que a função $f_{\tilde{\rho}}(\tilde{\rho})$ proposta neste trabalho apresenta um ganho menor para erros pequenos, proporcionando uma transição mais suave entre os valores positivos e negativos do erro de formação.

A Figura 2.5 ilustra o vetor de velocidades de referência $\dot{\xi}_{fr}$ gerado após a aplicação deste controlador. Neste caso, para um completo entendimento da ação do controlador, considerou-se uma variação nos parâmetros de formação desejados para o seguidor, ou seja, o vetor $\dot{\rho}_d$ é não nulo.

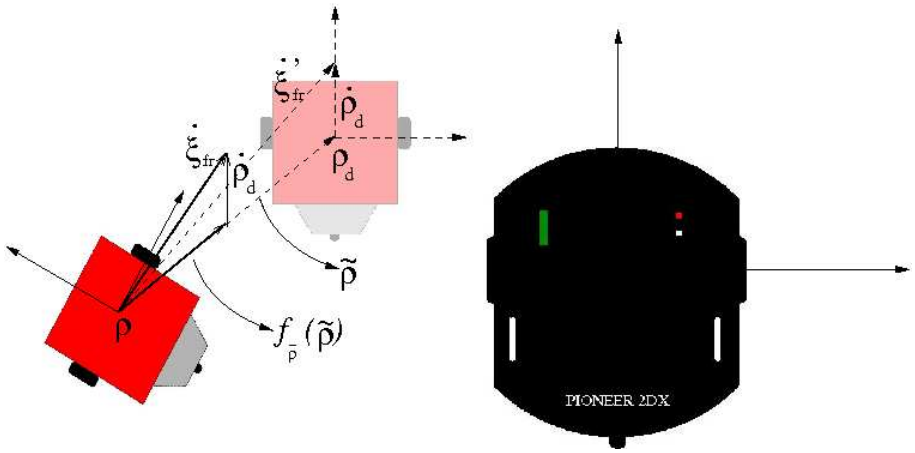


Figura 2.5: Resultado da aplicação do Controlador de Formação quando os parâmetros desejados do seguidor estão variando.

Nota-se claramente que o vetor $\dot{\xi}_{fr}$ possui uma orientação diferente daquela que teria se a função $f_{\tilde{\rho}}(\tilde{\rho})$ não fosse aplicada. Isso ocorre devido à saturação imposta por $f_{\tilde{\rho}}(\tilde{\rho})$ sobre o

erro de formação, $\tilde{\rho}$. Como $\dot{\rho}_d$ não sofre nenhum tipo de saturação, a soma $\dot{\rho}_d + \tilde{\rho}$, representada por $\dot{\xi}'_{fr}$, é diferente da soma $\dot{\rho}_d + f_{\tilde{\rho}}(\tilde{\rho}) = \dot{\xi}_{fr}$ em módulo e orientação. Entretanto, este desvio de orientação de $\dot{\xi}_{fr}$ não afeta a estabilidade do controlador, pois à medida que o seguidor se aproxima da postura desejada, $\|\tilde{\rho}\| \to 0$ e, portanto, $\dot{\xi}_{fr} \to \dot{\rho}_d$. Esse efeito, por sua vez, ocorrerá independente do uso da função $f_{\tilde{\rho}}(\tilde{\rho})$.

Na verdade, na maioria dos casos, durante uma navegação normal, a formação desejada não varia. Isso é mais comum durante, por exemplo, uma manobra de desvio de obstáculos. Assim, $\dot{\rho}_d$ é, na maior parte do tempo, um vetor nulo, o que elimina o desvio de orientação de $\dot{\xi}_{fr}$ durante uma navegação normal. Isto está ilustrado na Figura 2.6.

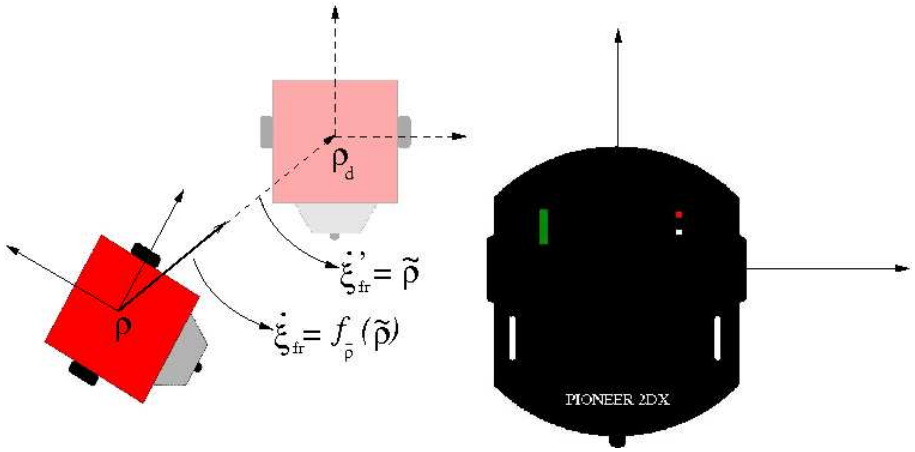


Figura 2.6: Resultado da aplicação do Controlador de Formação quando os parâmetros desejados do seguidor não variam.

Dessa forma, a função deste controlador é, então, gerar os sinais de velocidade que cada seguidor deve ter apenas para entrar em formação, ou seja, considera apenas os erros de formação, não levando em conta a orientação dos seguidores ou o movimento do líder. Em outras palavras, gera sinais de velocidade que conduzem os seguidores à formação desejada considerando estes robôs como holonômicos e supondo que o líder está parado. Embora não estejam de acordo com a realidade, estas considerações não são arriscadas, pois a ação do Controlador de Compensação, que ocorre em paralelo, as corrige, como será visto na próxima seção.

2.3 O Controlador de Compensação

Para que a equipe possa navegar por um ambiente, o líder deve possuir movimento. Este movimento é caracterizado por velocidades linear e angular. Logo, para alcançar e manter uma determinada formação, estas velocidades precisam ser compensadas pelos seguidores. Isso é feito somando-se ao vetor $\dot{\xi}_{fr}$ os vetores $\dot{\xi}_l$ e $\dot{\xi}_\omega$ que são, respectivamente, as compensações

das velocidades linear e angular do líder. Obtém-se, assim, o vetor velocidade de referência $\dot{\xi}_r$, dado pela Equação 2.8, o qual traduz as velocidades que devem ter os seguidores para ao mesmo tempo entrar em formação e compensar a translação e rotação do líder.

$$\dot{\xi}_r = \dot{\xi}_{fr} + \dot{\xi}_l + \dot{\xi}_\omega \tag{2.8}$$

Caso o líder possua apenas velocidade linear, o termo $\dot{\xi}_\omega$ pode ser desprezado. Assim, a compensação é conseguida simplesmente somando a velocidade do líder à componente de $\dot{\xi}_{fr}$ na direção y do seu referencial, como na Equação 2.9. Este é o efeito compensador de $\dot{\xi}_l$.

$$\dot{\xi}_r = \dot{\xi}_{fr} + \begin{pmatrix} 0 \\ v_l \\ \vdots \\ 0 \\ v_l \end{pmatrix} \tag{2.9}$$

onde v_l é a velocidade linear do líder.

Este efeito é ilustrado pela Figura 2.7, onde $\dot{\xi}_{frx}$ e $\dot{\xi}_{fry}$ são as componentes de $\dot{\xi}_{fr}$ nas direções x e y. Neste caso, a compensação é alcançada assintoticamente, desde que a velocidade imposta pelo líder seja menor que a máxima velocidade linear que os seguidores podem ter. Uma vez alcançada a postura desejada, o seguidor precisa apenas manter sua velocidade igual à do líder.

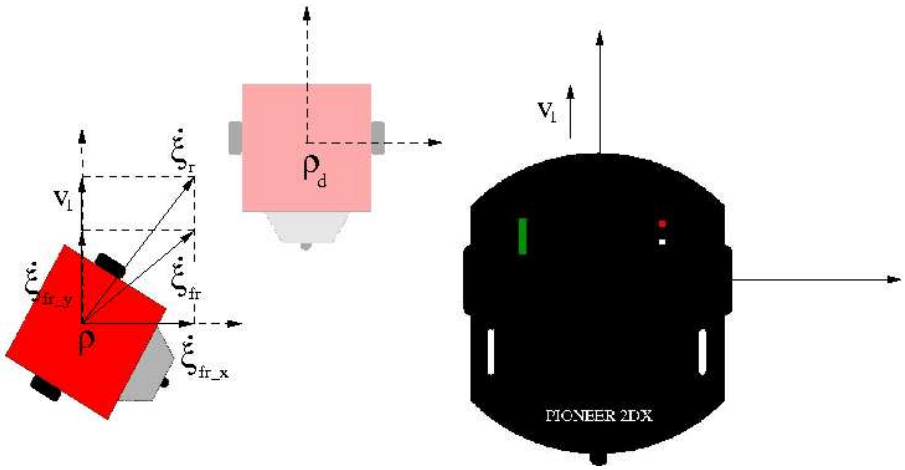


Figura 2.7: Efeito da compensação da velocidade linear do líder na determinação da velocidade de referência do seguidor.

Entretanto, normalmente o líder possui tanto velocidade linear quanto angular. Devido à sua rotação, esta compensação está diretamente ligada à geometria que descreve a trajetória dos

robôs da equipe. De acordo com a Figura 2.8, o líder e o seguidor descrevem raios iguais a, respectivamente, r e r_i. Considera-se aqui, por simplicidade, que o seguidor já se encontra na postura desejada. Sendo ω_l a velocidade angular do líder, para manter a formação, os seguidores devem ajustar suas velocidades de tal forma a descrever círculos concêntricos (se ω_l é constante) ou instantaneamente concêntricos (se ω_l está variando) com aquele descrito pelo líder.

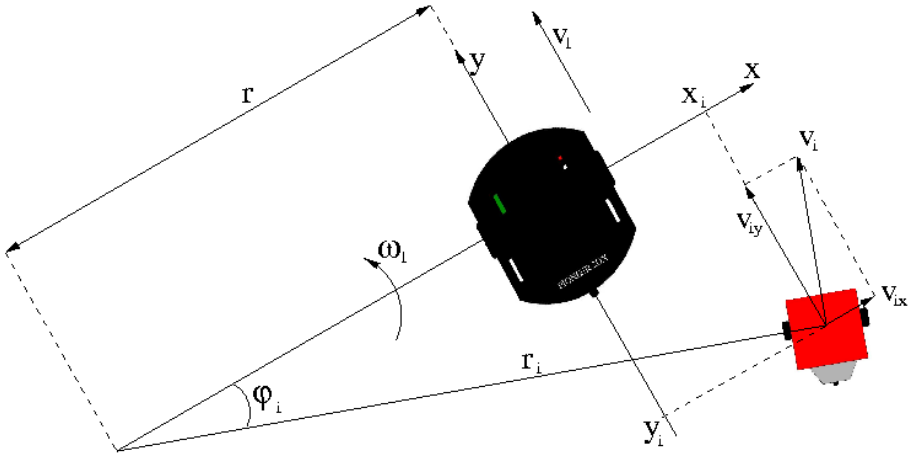


Figura 2.8: Geometria para a compensação simultânea das velocidades linear e angular do líder.

Assim, como v_l e ω_l são conhecidos, r e r_i são dados pelas Equações 2.10 e 2.11. Os vetores v_{ix} e v_{iy}, dados pelas Equações 2.14 e 2.15, são as componentes da velocidade linear de compensação do i-ésimo seguidor representadas no referencial do líder e expressam a soma $\dot{\xi}_l + \dot{\xi}_\omega$, ou seja, já incluem as contribuições de v_l e ω_l.

$$r = \frac{v_l}{\omega_l} \tag{2.10}$$

$$r_i = \sqrt{(r+x_i)^2 + (y_i)^2} \tag{2.11}$$

$$\varphi_i = arctan\left(\frac{y_i}{r+x_i}\right) \tag{2.12}$$

$$v_i = \omega_l r_i \tag{2.13}$$

$$v_{ix} = v_i \cos\left(\varphi_i + \frac{\pi}{2}\right) \tag{2.14}$$

$$v_{iy} = v_i \sin\left(\varphi_i + \frac{\pi}{2}\right) \tag{2.15}$$

Uma outra situação é quando o líder apresenta apenas rotação, isto é, possui apenas velocidade angular. Neste caso, é o termo $\dot{\xi}_l$ que é desprezado. Esta situação é exibida na Figura 2.9.

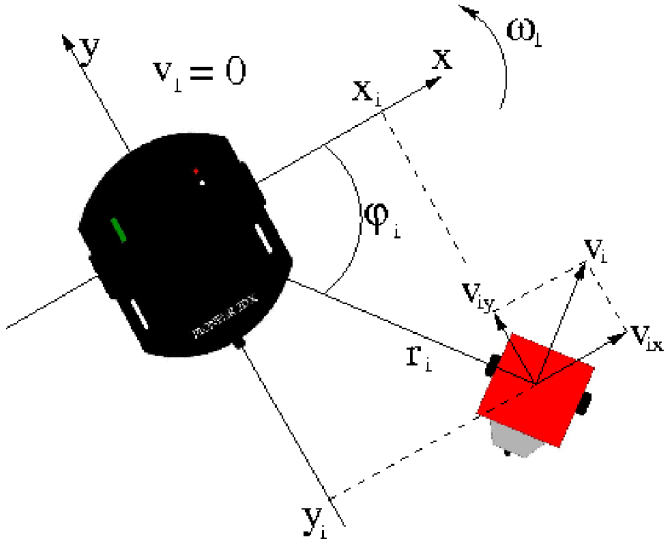


Figura 2.9: Geometria para compensação da velocidade angular do líder.

Neste caso, as Equações 2.10, 2.11 e 2.12 se reduzem a

$$r = 0 \tag{2.16}$$

$$r_i = \sqrt{(x_i)^2 + (y_i)^2} \tag{2.17}$$

$$\varphi_i = arctan\left(\frac{y_i}{x_i}\right) \tag{2.18}$$

e, portanto, a direção de v_i é perpendicular à r_i e v_{ix} e v_{iy} são obtidos conforme 2.14 e 2.15.

Uma vez conhecidas as compensações a serem feitas para alcançar a formação e o seguimento do líder, os comandos de velocidade linear e angular que devem ser enviados ao i-ésimo robô seguidor são dados pelas Equações 2.19 e 2.20, já que estes robôs são não-holonômicos e não podem realizar $\dot{\xi}_{ri}$ diretamente.

$$\dot{\xi}_{ci} = \|\dot{\xi}_{ri}\| \cos(\tilde{\alpha}_i) \tag{2.19}$$

$$\omega_{ci} = \dot{\alpha}_{ri} + f_{\tilde{\alpha}}(\tilde{\alpha}_i) + \omega_l \tag{2.20}$$

onde $\|\dot{\xi}_{ri}\|$ é a norma da velocidade de referência para o i-ésimo robô seguidor e $\dot{\alpha}_{ri}$ é a variação, no tempo, de sua orientação, α_{ri}. Portanto, α_{ri} é a orientação desejada para o i-ésimo seguidor, que tem sua orientação atual representada por α_i. Logo, o termo $\tilde{\alpha}_i = \alpha_{ri} - \alpha_i$ é o erro angular do i-ésimo seguidor. Todos esses ângulos estão ilustrados na Figura 2.10, onde o subscrito foi omitido por razões de clareza.

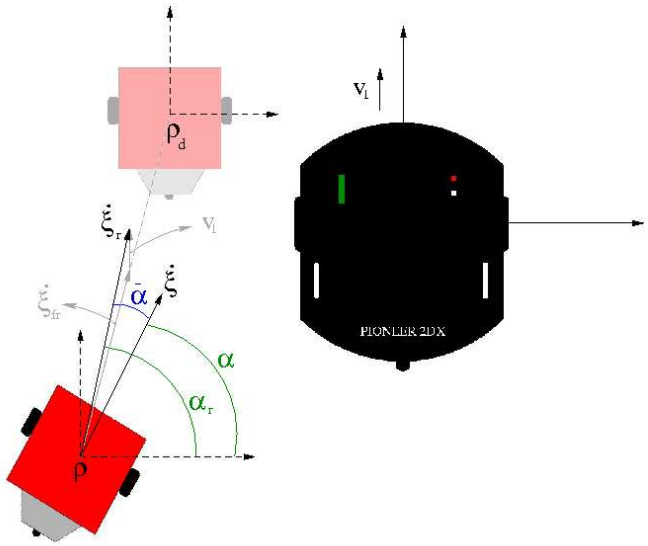


Figura 2.10: Ângulos relacionados à determinação das velocidades linear e angular de comando de um seguidor.

A função $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$, assim como no caso do controlador de formação, é uma função de saturação sobre o erro. Em [35], esta função é definida segundo a Equação 2.21.

$$f_{\tilde{\alpha}}(\tilde{\alpha}_i) = k_\omega tanh(\tilde{\alpha}_i) \tag{2.21}$$

onde k_ω representa o valor de saturação sobre erro de orientação. A função $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ tem a finalidade de evitar que erros grandes de orientação (geralmente iniciais) gerem velocidades angulares de comando muito altas. Isto poderia comprometer a estabilidade e submeter os motores dos robôs a variações abruptas de tensão. A forma de $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ está ilustrada na Figura 2.11.

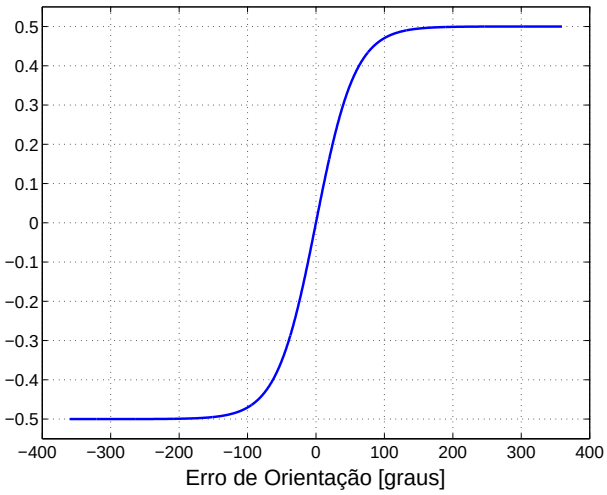


Figura 2.11: Aparência da função $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ para $k_\omega = 0.5$.

Entretanto, devido à sua forma, $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ pode causar oscilações na velocidade angular dos seguidores quando os erros de orientação são pequenos. Assim, mesmo que um determinado seguidor esteja em sua posição desejada, sua trajetória pode ser oscilatória. Logo, propõe-se aqui uma pequena modificação nesta função, a fim de tornar a transição entre valores positivos e negativos do erro de orientação mais suave. Neste caso, a nova função exibe uma inclinação que tende a zero à medida que o módulo do erro diminui. A nova $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ está definida na Equação 2.22 e sua forma pode ser vista na Figura 2.12.

$$f_{\tilde{\alpha}}(\tilde{\alpha}_i) = k_{\omega 1} tanh^3(k_{\omega 2}\ \tilde{\alpha}_i) \tag{2.22}$$

Desta vez, o valor de saturação é determinado por $k_{\omega 1}$, enquanto $k_{\omega 2}$ controla a inclinação da função, ou seja, o quão rápido $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ atinge a saturação. Assim, mais uma vez, a função $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ permite maior controle sobre os parâmetros que influenciam seu comportamento e, portanto, o comportamento do controlador.

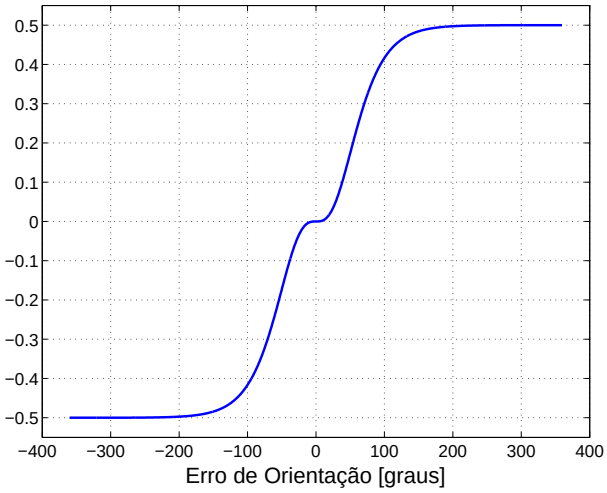


Figura 2.12: Aparência da nova função $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$ para $k_{\omega 1} = 0.5$ e $k_{\omega 2} = 1$.

Vale lembrar que as modificações feitas nas equações do controlador, embora melhorem seu desempenho, aumentam o custo computacional envolvido. Por outro lado, este aumento não é significativo se comparado ao custo do processamento das imagens omnidirecionais. Portanto, as modificações realizadas não afetam o desempenho do sistema como um todo.

O diagrama de blocos da Figura 2.13 ajuda a compreender o funcionamento do controlador empregado. Nesse diagrama, o grande retângulo cinza representa o controlador. Percebe-se que as variáveis de entrada são o vetor com os parâmetros de formação desejados e as velocidades linear e angular do líder. O diagrama também deixa claro que os controladores de Formação e Compensação trabalham em paralelo e que a combinação das suas respostas fornece o vetor com as velocidades de referência para os seguidores, $\dot{\xi}_r$. Este, juntamente com a velocidade angular

do líder e os vetores $\dot{\alpha}_r$ e $\tilde{\alpha}$ são usados para gerar as velocidades linear e angular que cada seguidor deve ter para alcançar e manter a formação desejada, ou seja, são usados para gerar os sinais de controle. O vetor $\dot{\alpha}_r$ representa a derivada temporal das orientações das velocidades de referência de cada seguidor, enquanto o vetor $\tilde{\alpha}$ reúne os erros angulares de cada um dos seguidores. O sistema de visão captura imagens da cena formada pelos robôs da equipe. O processamento dessas imagens fornece as posturas de todos os seguidores, representadas pelo vetores ρ e α, que fazem a realimentação do controlador.

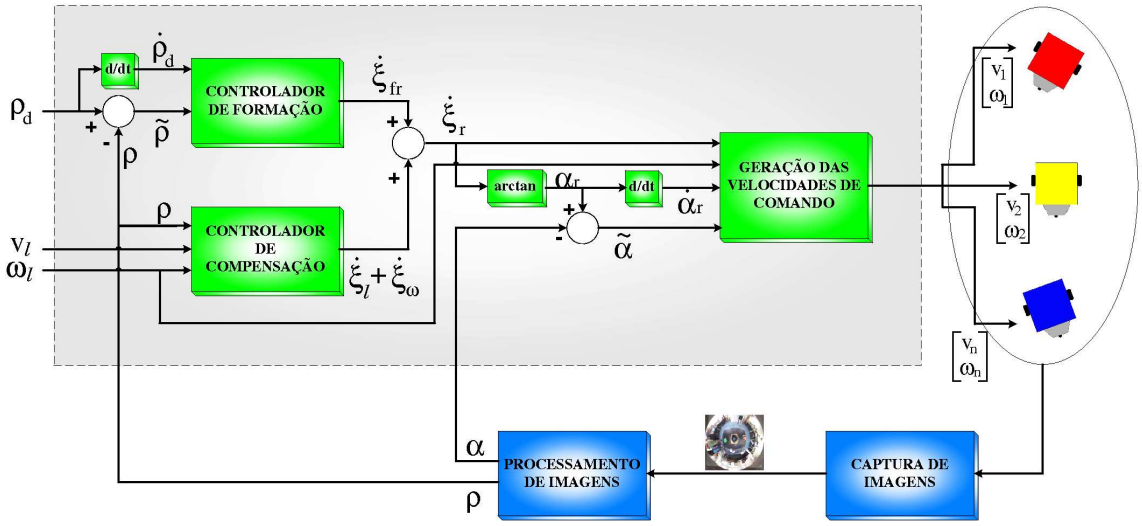


Figura 2.13: Diagrama de blocos do controlador não-linear utilizado.

2.4 Prova de Estabilidade do Controlador

2.4.1 Prova para o controlador de formação

Os robôs seguidores, devido às suas dinâmicas, não podem alcançar instantaneamente as velocidades de formação desejadas. No entanto, este processo ocorre assintoticamente desde que a velocidade do líder seja menor que a máxima velocidade que podem desenvolver. Isso está representado pela Equação 2.23.

$$\dot{\xi} \rightarrow \dot{\xi}_{fr} \tag{2.23}$$

onde $\dot{\xi}$ é o vetor atual de velocidades e $\dot{\xi}_{fr}$ é o vetor de velocidades de referência para a formação. A Equação 2.23 também pode ser escrita como na Equação 2.24.

$$\dot{\xi}_{fr} = \dot{\xi} + \eta \qquad \text{com} \qquad \|\eta\| \rightarrow 0 \tag{2.24}$$

onde η traduz os erros nas velocidades de formação. A lei de formação é dada pela Equação 2.25, como discutido no início deste capítulo.

$$\dot{\xi}_{fr} = J^{-1}(\xi)\left(\dot{\rho}_d + f_{\tilde{\rho}}(\tilde{\rho})\right) \tag{2.25}$$

Substituindo 2.24 em 2.25 obtém-se a Equação 2.26.

$$\dot{\xi} + \eta = J^{-1}(\xi)\left(\dot{\rho}_d + f_{\tilde{\rho}}(\tilde{\rho})\right) \tag{2.26}$$

A multiplicação de ambos os lados da Equação 2.26 por $J(\xi)$ leva à Equação 2.27.

$$J(\xi)\dot{\xi} + \eta_1 = \dot{\rho}_d + f_{\tilde{\rho}}(\tilde{\rho}) \qquad onde \qquad \eta_1 = J(\xi)\eta \tag{2.27}$$

Como se sabe, $\dot{\rho} = J(\xi)\dot{\xi}$ (Equação 2.4), o que conduz à Equação 2.28.

$$\dot{\rho} + \eta_1 = \dot{\rho}_d + f_{\tilde{\rho}}(\tilde{\rho}) \tag{2.28}$$

A derivada de $\tilde{\rho}$ produz a Equação 2.29.

$$\dot{\tilde{\rho}} = \dot{\rho}_d - \dot{\rho} \qquad \Rightarrow \qquad \dot{\rho}_d = \dot{\tilde{\rho}} + \dot{\rho} \tag{2.29}$$

A substituição da Equação 2.29 na Equação 2.28 resulta em

$$\dot{\tilde{\rho}} = \eta_1 - f_{\tilde{\rho}}(\tilde{\rho}). \tag{2.30}$$

Em seguida, propõe-se a seguinte função candidata de Lyapunov:

$$V = \frac{1}{2}\tilde{\rho}^T\tilde{\rho}, \tag{2.31}$$

cuja derivada temporal é

$$\dot{V} = \tilde{\rho}^T\dot{\tilde{\rho}}. \tag{2.32}$$

Substituindo a Equação 2.30 na Equação 2.32, obtém-se

$$\dot{V} = \tilde{\rho}^T\eta_1 - \tilde{\rho}^T f_{\tilde{\rho}}(\tilde{\rho}). \tag{2.33}$$

Para que $\dot{V}$ seja definida negativa é necessário que

$$\frac{\|k_f(\tilde{\rho})\|}{a + \|\tilde{\rho}\|}\|\tilde{\rho}\|^2 > \|\eta_1\|\|\tilde{\rho}\|, \tag{2.34}$$

onde $\|k_f(\tilde{\rho})\|$ é definida como

$$\|k_f(\tilde{\rho})\| = k_{f1} + k_{f2}tanh(\|\tilde{\rho}\|) \tag{2.35}$$

e, portanto,

$$\|\tilde{\rho}\| > \frac{a\|\eta_1\|}{k_{f1} + k_{f2} tanh(\|\tilde{\rho}\|) - \|\eta_1\|}. \tag{2.36}$$

Como os seguidores atingem as velocidades de formação, $\|\eta\| \to 0$. Assim, $\|\eta_1\| \to 0$ e a condição imposta pela Equação 2.36 será verdadeira para algum tempo finito. Tão logo esta condição se torne verdadeira, $\dot{V}$ será definida negativa, o que significa que $\|\tilde{\rho}(t)\| \to 0$ com $t \to \infty$.

2.4.2 Prova para as leis de controle (comandos) para os robôs seguidores

A variação temporal da orientação relativa dos seguidores é expressa pela Equação 2.37.

$$\dot{\alpha} = \omega_c - \omega_l \tag{2.37}$$

onde a velocidade angular de comando, ω_c, é dada por

$$\omega_c = \dot{\alpha}_r + f_{\tilde{\alpha}}(\tilde{\alpha}) + \omega_l. \tag{2.38}$$

Substituindo a Equação 2.38 na Equação 2.37, tem-se

$$\dot{\alpha} = \dot{\alpha}_r + f_{\tilde{\alpha}}(\tilde{\alpha}) \tag{2.39}$$

A derivada do erro angular, $\tilde{\alpha}$, em relação ao tempo, resulta em

$$\dot{\tilde{\alpha}} = \dot{\alpha}_r - \dot{\alpha} \qquad \Rightarrow \qquad \dot{\alpha}_r = \dot{\tilde{\alpha}} + \dot{\alpha}. \tag{2.40}$$

Com isso, a Equação 2.39 pode ser reescrita como

$$\dot{\tilde{\alpha}} + f_{\tilde{\alpha}}(\tilde{\alpha}) = 0. \tag{2.41}$$

Assim, propõe-se a seguinte função candidata de Lyapunov:

$$V = \frac{1}{2}\tilde{\alpha}^T\tilde{\alpha}, \tag{2.42}$$

cuja derivada temporal é

$$\dot{V} = \tilde{\alpha}^T\dot{\tilde{\alpha}} = -\tilde{\alpha}^T f_{\tilde{\alpha}}(\tilde{\alpha}). \tag{2.43}$$

Como $f_{\tilde{\alpha}}(\tilde{\alpha})$ é uma função ímpar, tem-se que $\tilde{\alpha}^T f_{\tilde{\alpha}}(\tilde{\alpha}) > 0$ para $\tilde{\alpha} \neq 0$, conclui-se que $\dot{V}$ é definida negativa ($\dot{V} < 0$). Portanto $\|\tilde{\alpha}(t)\| \to 0$ para $t \to \infty$.

Aproveitando este resultado, conclui-se também que $\dot{\xi}_{ci} \to \|\dot{\xi}_{ri}\|$, pois $\cos(\tilde{\alpha}_i) \to 1$.

Com isso, fica demonstrado que os erros de formação e seguimento tendem assintoticamente a zero, justificando a adoção destas leis de controle.

2.5 Conclusões

Este capítulo apresentou o controlador utilizado neste trabalho, seu funcionamento e suas características. Trata-se de um controlador de postura, que aproveita a não linearidade das equações que descrevem o sistema para gerar sinais de controle que conduzem a equipe à formação desejada de forma estável. Para tanto, divide-se em dois controladores menores, onde o primeiro, chamado Controlador de Formação, tem a tarefa de produzir os sinais de controle referentes ao sistema de coordenadas do líder que levam os robôs seguidores às posturas esperadas. Porém, como toda a equipe está em movimento, inclusive o líder, o sistema de coordenadas deste também se move. Considerar e compensar este movimento é responsabilidade do segundo controlador, o Controlador de Compensação, que leva em conta a geometria da trajetória de cada robô da equipe em suas equações.

Para controlar as posturas dos seguidores, o líder precisa estimar cada uma delas, o que é feito através de realimentação visual, usando o sistema omnidirecional que possui. Este é o assunto do próximo capítulo, que descreve as técnicas de processamento de imagens empregadas neste trabalho.

3 O Processamento de Imagens

3.1 Introdução

O processamento de imagens vem sendo usado há alguns anos nos mais variados tipos de aplicação. A utilização de imagens representa, hoje, um papel fundamental em sistemas de comunicação, segurança, medicina, tecnologias de apoio a pessoas com deficiência [47, 48], entretenimento, controle de qualidade e processos, aplicações militares e, é claro, robótica.

A utilização de imagens na robótica vem se tornando cada vez mais marcante. Isto se deve ao fato de que imagens são capazes de fornecer, de uma só vez, grande quantidade de informação a respeito do ambiente de trabalho, como a localização de outros agentes, obstáculos ou pessoas. Com o avanço da tecnologia, a miniaturização, a eficiência e a qualidade dos sistemas de visão aumentam a cada ano. Além disso, o surgimento de ferramentas computacionais e o aumento da capacidade de processamento impulsionam ainda mais a expansão da visão artificial em aplicações da robótica.

Alguns trabalhos que utilizam a visão artificial em cooperação robótica podem ser citados, como [25, 26, 28, 33, 36, 49]. Em [25], os autores introduzem o conceito de estruturas virtuais e utilizam controle bidirecional para controlar a formação de um grupo de robôs móveis. A localização e a deteminação da orientação destes são feitas através de visão. Simulações e experimentos com robôs reais foram realizados. Porém, o sistema de visão está fixo em um ponto acima dos robôs, limitando sua área de trabalho. Da mesma forma, os trabalhos [33, 36, 49] utilizam câmaras fixas para controlar os robôs.

Em [28], imagens omnidirecionais são usadas para se obter cooperação entre os integrantes de um grupo de robôs móveis. Os autores desenvolveram um cenário no qual cada seguidor utiliza fluxo ótico para estimar as posições relativas dos outros robôs, permitindo que o grupo mantenha, visualmente, a formação desejada sem a necessidade de segmentação por cores ou comunicação entre eles. Por outro lado, o custo computacional envolvido no cálculo do fluxo ótico é alto e os resultados são mostrados apenas através de simulações.

Outro trabalho que utiliza imagens omnidirecionais em benefício da cooperação é [26], onde cada robô possui seu próprio sistema catadióptrico de visão, permitindo a adoção de uma estratégia de controle descentralizado e eliminando a necessidade de comunicação entre os robôs. Entretanto, todo o processamento das imagens é feito em um computador central e externo aos robôs. Além disso, assim como em [28], o uso de vários sistemas omnidirecionais eleva o custo da equipe.

Diferentemente de [26, 28] e como já exposto anteriormente, este trabalho propõe uma arquitetura de controle centralizado, no qual o sistema de visão está embarcado apenas no robô líder. Com isso, o sistema se move junto com a equipe e, portanto, a área de trabalho não é limitada, como acontece em [25, 33, 36, 49]. Assim, para fornecer ao controlador proposto neste trabalho as posturas dos seguidores de forma adequada, é necessário processar as imagens omnidirecionais capturadas pelo sistema de visão. Para isso, técnicas de processamento de imagens disponíveis na literatura foram utilizadas.

Este capítulo dedica-se, portanto, a descrever as técnicas de processamento de imagens aplicadas às imagens omnidirecionais para garantir a realimentação visual para o controlador, fornecendo a postura de cada seguidor.

3.2 O Processamento das Imagens Omnidirecionais

Como exposto no Capítulo 1, imagens omnidirecionais, cuja finalidade é ampliar o campo de visão, podem ser conseguidas de várias formas; uma delas é através da combinação de espelhos convexos e câmaras de vídeo. Por outro lado, esta ampliação traz consigo efeitos indesejados, a saber, a perda de resolução das imagens e a distorção não linear. O primeiro destes efeitos ocorre porque toda câmara de vídeo tem um número limitado de *pixels* em seu CCD [1]. Quando, através da reflexão no espelho, toda a área em torno do sistema de visão é mapeada neste CCD, cada *pixel* deve corresponder a uma área relativamente grande no mundo, causando a perda de resolução. Em outras palavras, com o uso do espelho, mapeia-se uma área muito maior na mesma quantidade de *pixels*.

O segundo efeito, a distorção, acontece porque a projeção de um ponto qualquer do mundo no plano da imagem obedece a uma equação não linear que depende da curvatura do espelho. Assim, à medida que um objeto se afasta do sistema de visão, a distorção aumenta rapidamente. Dependendo do tamanho do objeto de interesse, sua detecção pode se tornar impraticável a poucos metros do sistema.

[1] Do inglês *Charge Coupled Device*, ou Dispositivo de Carga Acoplada, é o sensor fotossensível presente nas câmaras de vídeo, máquinas fotográficas digitais e *scanners*, responsável pela captura da imagem.

Entretanto, estes efeitos são superados pela vantagem de se obter um campo visual horizontal de 360^o. Com este campo, o líder é capaz de localizar todos os integrantes da equipe capturando apenas uma imagem, que também pode ser usada para desvio de obstáculos e mapeamento do ambiente sem a necessidade de rotacionar a câmara em torno de um eixo vertical.

Para que a equipe possa navegar por um ambiente em determinada formação, o líder deve conhecer a postura de cada robô celular pertencente ao grupo. Entretanto, quando o sistema é inicializado, o líder não conhece as posturas e cores de seus seguidores. Deste modo, é necessário, antes de iniciar a navegação, detectar a posição inicial, a cor e a orientação inicial de cada robô celular integrante da equipe. Uma vez encerrada esta etapa, o líder inicia, então, a navegação.

Assim, o processamento das imagens pode ser dividido em três etapas, que são:

1. Detecção da posição inicial;
2. Rastreamento (*Tracking*) para determinação da orientação inicial;
3. Rastreamento para controle de formação.

A implementação destas etapas contou com a utilização da biblioteca de processamento de imagens OpenCV (*Open Source Computer Vision Library*) [50]. Trata-se de uma biblioteca gratuita disponibilizada pela Intel, com código aberto, bem documentada e testada.

Como já foi dito no capítulo anterior, o controlador deve ser independente do sistema de visão empregado. Uma maneira de fazer isto e ao mesmo tempo eliminar a distorção das imagens é remapeá-las para imagens do tipo *bird's eye view* [51, 52], cuja aparência imita uma projeção ortográfica do ambiente visto de cima. Infelizmente, este remapeamento depende de uma calibração do sistema e representa maior custo no processamento das imagens.

Isso motivou a definição da transformação Γ, que não depende de calibração do sistema, possui baixo custo computacional e cujas funções que a compõem são fáceis de ser obtidas. As informações necessárias para sua construção são as coordenadas do centro da imagem omnidirecional e as tabelas que relacionam as posições na imagem e no mundo. Embora o número de funções que compõem esta transformação possa ser elevado, aumentando sua precisão, neste trabalho, Γ foi definida usando-se um conjunto de quatro funções polinomiais. Quatro tabelas relacionando as distâncias no plano da imagem com as distâncias no mundo real foram construídas e em seguida cada uma delas foi usada para interpolar uma função polinomial. A área de trabalho ao redor do líder foi dividida em quatro setores circulares, como na Figura 3.1, e a cada setor estão associadas uma tabela e uma função.

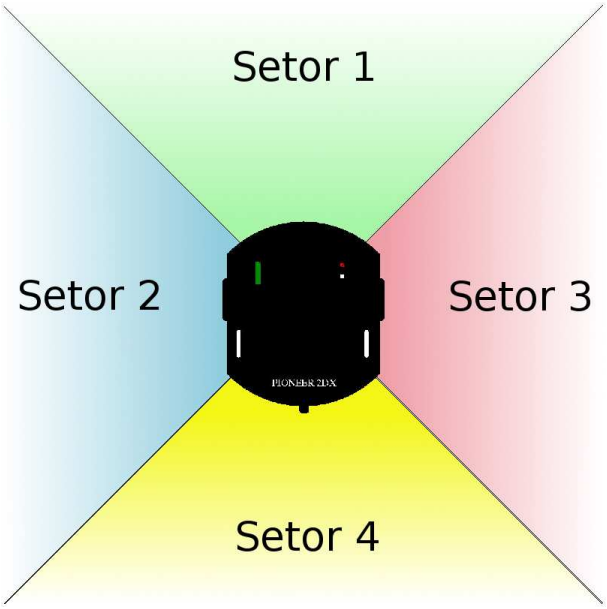


Figura 3.1: Divisão em setores circulares da área de trabalho para a obtenção das funções polinomiais que definem a transformação Γ.

As funções assim obtidas são mostradas na Figura 3.2. Dessa forma, um determinado seguidor terá sua posição convertida do plano da imagem para o mundo (plano do chão) de acordo com o setor onde se encontra.

Vale ressaltar que esta abordagem é significativamente mais rápida que o remapeamento *bird's eye view*.

3.2.1 Detecção da Posição Inicial

A forma do espelho hiperbólico utilizado neste trabalho para formar as imagens omnidirecionais é adequada para aplicações onde se deseja visualizar todo o ambiente ao redor do sistema, como pode ser visto na Figura 3.3. Entretanto, neste trabalho, a área de interesse está restrita a cerca de 2 metros em torno do robô líder. Isso ocorre devido à forma do espelho, que impõe forte distorção, e da câmara utilizada, incapaz de focá-lo se seu *zoom* for aumentado ou se colocada mais próxima. Qualquer objeto com dimensões próximas das dos robôs celulares, se situado a mais de 2 metros do sistema de visão, torna-se muito pequeno na imagem, comprometendo sua detecção. Logo, uma das considerações feitas neste trabalho é que todos os seguidores estarão sempre a uma distância de, no máximo, 2 metros do líder. Para detectar as posições iniciais dos seguidores, o líder deve ser capaz de focar sua atenção na área de interesse, também chamada, aqui, de área de trabalho. Para isso, foi utilizada uma máscara circular, que consiste em uma imagem binária onde um círculo branco, cujo centro tem as mesmas coordenadas do centro da imagem omnidirecional, define a região de interesse. Alterando-se o raio

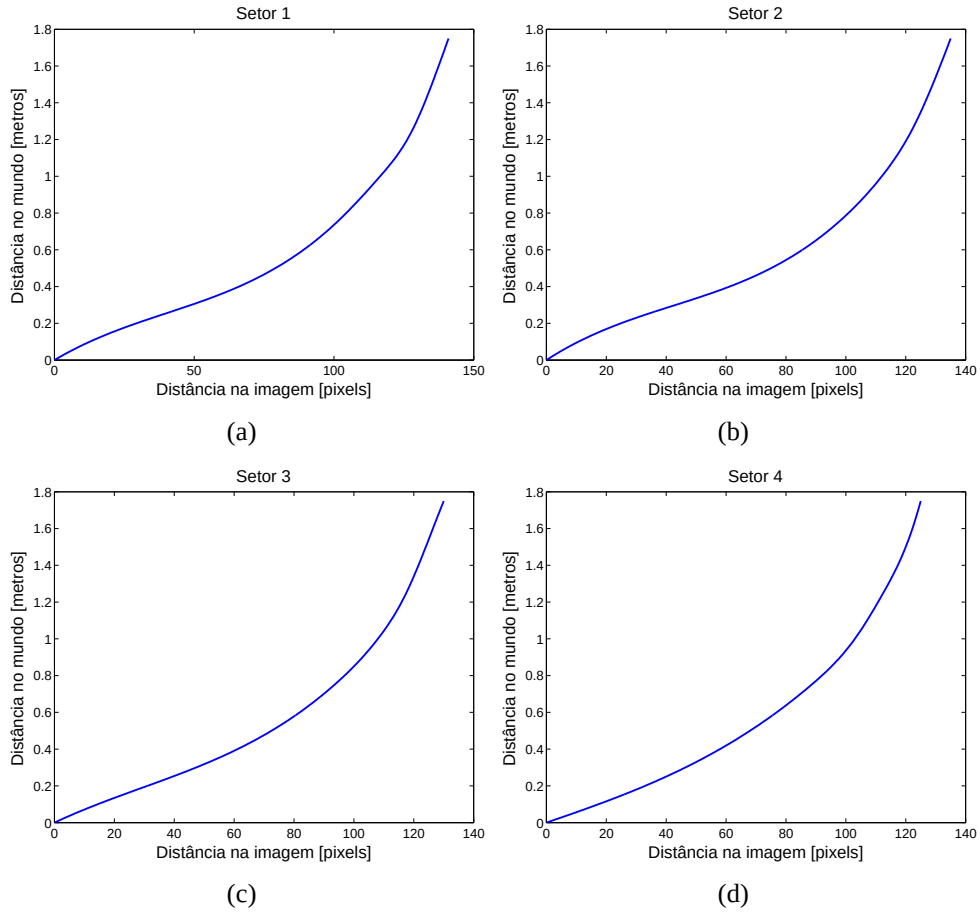


Figura 3.2: Gráficos exibindo o comportamento de cada função polinomial obtida. (a) Setor 1 (b) Setor 2 (c) Setor 3 e (d) Setor 4.

do círculo, altera-se a região de interesse. A máscara, que pode ser vista na Figura 3.4-(a), é aplicada à imagem através da operação E lógico, ou seja, serão mantidos apenas aqueles *pixels* que são diferentes de zero. O resultado pode ser visto na Figura 3.4-(b).

Cabe, aqui, uma observação importante: a limitação da área de trabalho é necessária, como foi dito, devido à forma do espelho e câmara usada. Juntamente com a exclusão da parte mais central das imagens (ocupada pela câmara e pelo líder), essa limitação faz da área útil para o processamento das imagens uma coroa circular que representa apenas $12,26\%$ da área total disponível. Sem dúvida, isso reduz a precisão da estimativa de postura dos seguidores, afetando o desempenho do controlador. Porém, deve-se ressaltar que esse inconveniente não ocorre devido ao emprego de imagens omnidirecionais, mais sim devido à câmara utilizada e à forma do espelho, que não é a mais indicada para a aplicação em questão.

Uma vez definida a área de trabalho, o passo seguinte é a detecção da posição inicial de cada seguidor. Para isso, foi utilizada a técnica de segmentação de movimento conhecida como *background subtraction*. Em linhas gerais, esta técnica consiste em, primeiramente, construir uma

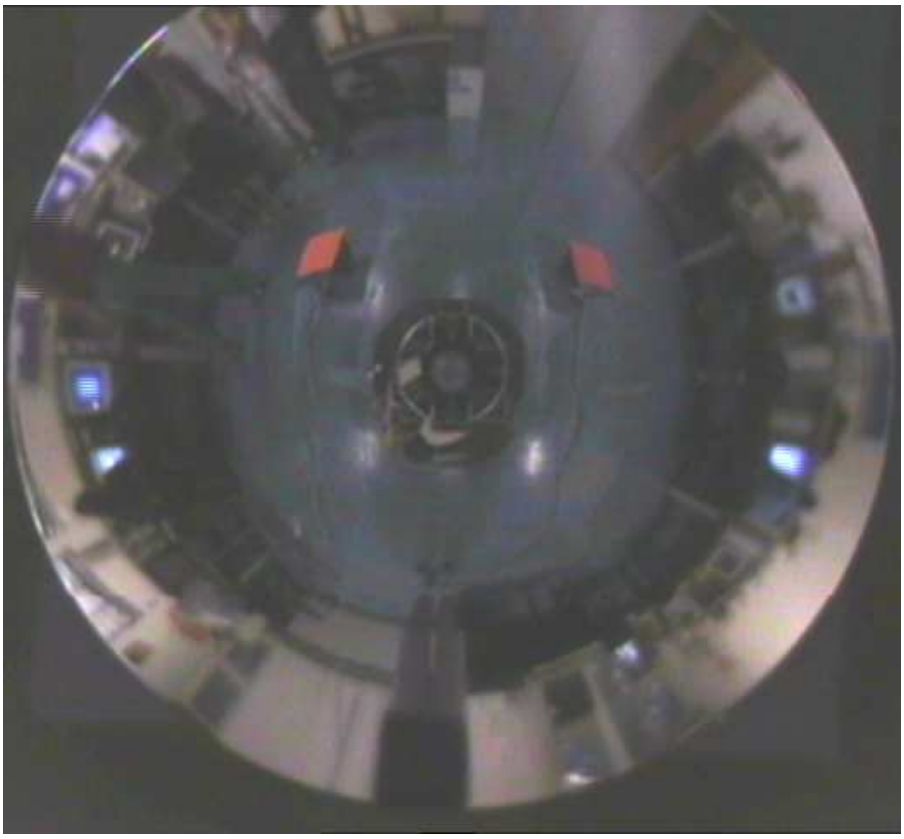
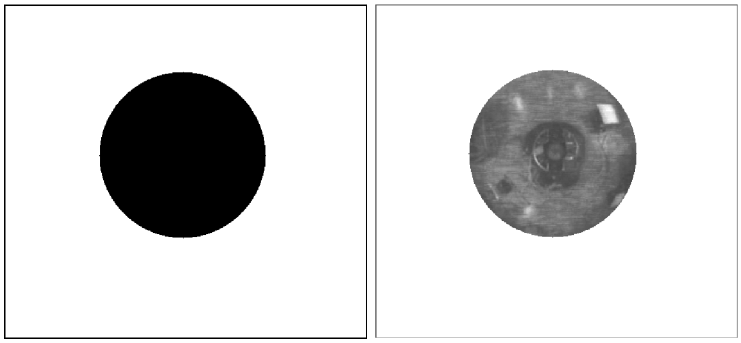


Figura 3.3: Imagem capturada pelo sistema de visão utilizado: a forma do espelho não é a mais indicada para este trabalho.



(a) (b)

Figura 3.4: (a) Máscara binária aplicada à imagem omnidirecional – imagem invertida para melhor visualização – e (b) resultado da aplicação da máscara, definindo a área de interesse para a detecção das posições iniciais dos seguidores.

imagem de fundo, ou *background*, que nada mais é do que a média de uma seqüência de imagens de uma cena estática. As imagens capturadas em seguida são, então, comparadas (através de subtração) com a imagem de fundo e as regiões que apresentarem diferenças significativas são segmentadas como regiões onde houve movimento.

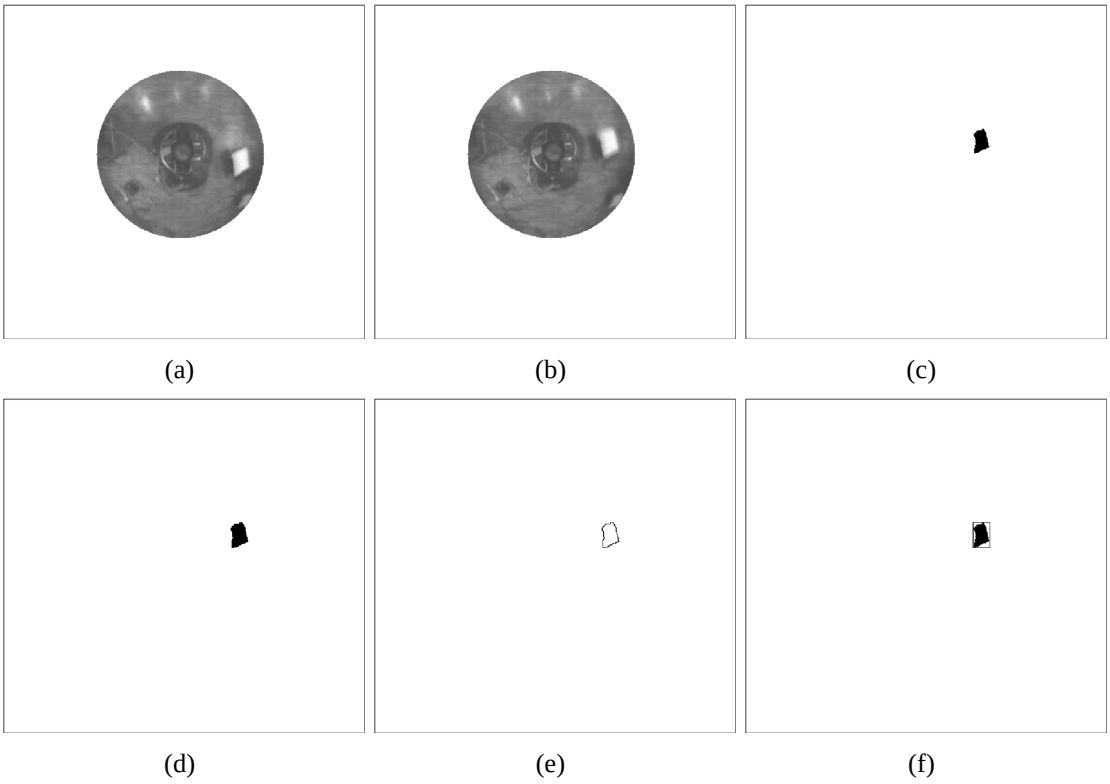
Embora seja possível o uso de imagens coloridas, esta técnica é normalmente implementada utilizando-se apenas imagens em tons de cinza, o que é suficiente para detectar movimento. Então, neste trabalho, cada imagem capturada, pertencente ao espaço de cores RGB (*Red, Green, Blue*), é convertida para o espaço HSV (*Hue, Saturation, Value*) [53]. Como o canal *Value* é a versão em tons de cinza da imagem capturada, somente este é utilizado para implementar o *background subtraction*.

Entretanto, devido à baixa resolução típica de imagens omnidirecionais e da presença de ruídos provenientes da iluminação e qualidade da câmara utilizada, foi implementada, neste trabalho, uma versão mais robusta deste método. Neste caso, é feita uma subtração entre dois *backgrounds*.

Dessa forma, a primeira tarefa do líder é a construção do primeiro *background*, chamado *background* base (Figura 3.5-(a)), que é feita durante 2 segundos. Durante este tempo, todos os robôs, inclusive o líder, permanecem parados. Em seguida, o líder ordena a um dos seguidores que se mova para frente, em linha reta, durante 1,5 segundos. Tão logo o seguidor termina seu movimento, um novo *background*, chamado *background* discriminante, é construído (Figura 3.5-(b)) e comparado com o primeiro. A diferença entre eles passa, então, por um processo de binarização, ou *thresholding*, mostrado na Figura 3.5-(c). O resultado é uma imagem binária que indica o movimento realizado pelo seguidor. A região segmentada, correspondente ao movimento, é comumente conhecida como *blob*.

O passo seguinte é a aplicação de uma operação morfológica, a dilatação, cujo elemento estruturante possui apenas 1 *pixel*. O resultado é colocado em uma nova imagem binária, mostrada na Figura 3.5-(d). Subtraindo a imagem na Figura 3.5-(c) da imagem na Figura 3.5-(d), é possível extrair a borda do *blob*, que aparece na Figura 3.5-(e). Por fim, determina-se o retângulo que circunscreve esta borda. Este é necessário por dois motivos: primeiro, assume-se que a posição do seguidor é aquela dada pelas coordenadas do centróide do retângulo; segundo, o algoritmo que realiza o rastreamento (ou *tracking*) baseado na segmentação por cores – e usado nas etapas seguintes do processamento das imagens omnidirecionais – depende deste retângulo para funcionar. Assim, foi desenvolvido um algoritmo para determinar tal retângulo, e o resultado de sua aplicação pode ser visto na Figura 3.5-(f).

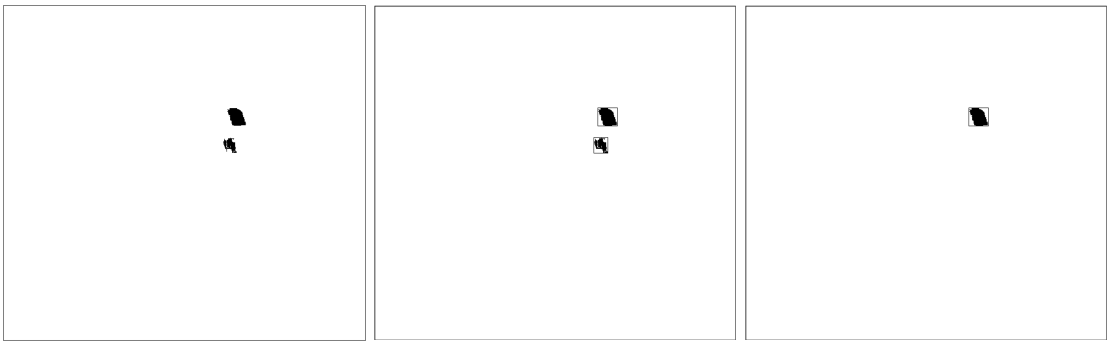
Entretanto, devido à presença dos ruídos já citados, nem sempre todo este processamento resulta em uma imagem como na Figura 3.5-(f). Às vezes, mais de um *blob* pode aparecer, como na Figura 3.6-(a). O processamento acima descrito resulta na imagem mostrada na Figura 3.6-(b), que confundiria o líder a respeito da localização do seguidor.



(a) (b) (c)

(d) (e) (f)

Figura 3.5: (a) *Background* base (b) *Background* discriminante (c) resultado da binarização após subtração entre os *backgrounds* (d) versão dilatada da imagem anterior (e) detecção da borda do *blob* e (f) resultado da aplicação do algoritmo que encontra o retângulo circunscrito.



(a) (b) (c)

Figura 3.6: (a) Imagem com mais de um *blob* detectado (b) resultado da detecção dos retângulos circunscritos e (c) imagem filtrada.

Faz-se necessário, então, um processo de filtragem que funciona da seguinte forma: primeiro, o canal *Hue*, que responde pelas informações de cor da imagem capturada é inicializado. Então, a cor predominante de cada *blob* é determinada. Isso é feito calculando-se o histograma de cada *blob* e, em seguida, identificando a cor que corresponde ao máximo valor do histograma. A Figura 3.7 exibe um histograma cuja cor predominante é a vermelha.

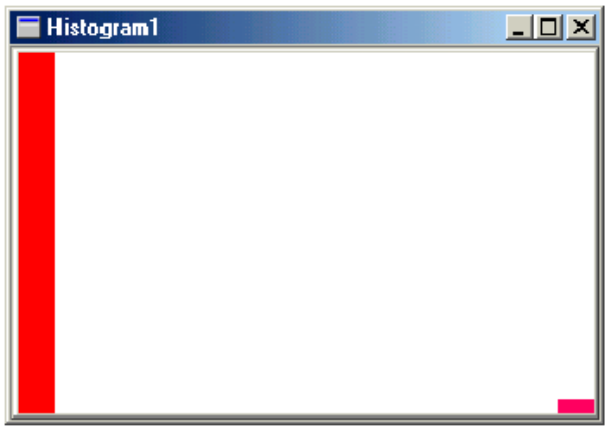


Figura 3.7: Histograma correspondente a um *blob* de cor vermelha.

Caso a distância entre os centróides dos retângulos circunscritos a cada *blob* seja menor que 20 cm (no plano do chão) e a cor predominante for a mesma, os retângulos são fundidos. Caso contrário, aquele que possuir a menor área será eliminado. O resultado pode ser visto na Figura 3.6-(c).

O procedimento descrito acima é usado para detectar a posição inicial de um seguidor. O mesmo processo deve ser repetido com os demais robôs. Entretanto, a criação de dois novos *backgrounds* não é necessaria, pois o *background* discriminante relativo a um seguidor pode ser usado como *background* base para o próximo seguidor a ser detectado. Assim, para o caso de n seguidores, ao invés de calcular $2n$ *backgrounds*, apenas $n+1$ serão necessários, reduzindo o tempo dispendido na detecção das posições iniciais.

A aplicação das técnicas de processamento de imagens (como *background subtraction*, binarização, dilatação) e dos algoritmos desenvolvidos (determinação do retângulo circunscrito e filtragem dos *blobs*) conferiu robustez à etapa de detecção das posições iniciais dos seguidores. O fluxograma da Figura 3.8 ajuda a compreender melhor os passos e respectivos resultados desta etapa.

Antes de passar para a próxima fase, denominada “Rastreamento para Determinação da Orientação Inicial”, é necessário identificar a cor de cada seguidor, pois um algoritmo baseado em segmentação por cores será usado não só na próxima, mas também na última etapa, para estimar as posturas dos seguidores, objetivo maior do processamento de imagens deste trabalho.

A detecção de cor já foi utilizada pelo algoritmo de filtragem de *blobs*, uma vez que identifica a cor predominante de cada *blob*. Por outro lado, esta informação é usada apenas para eliminar *blobs* indesejados, não sendo associada ao seguidor em questão.

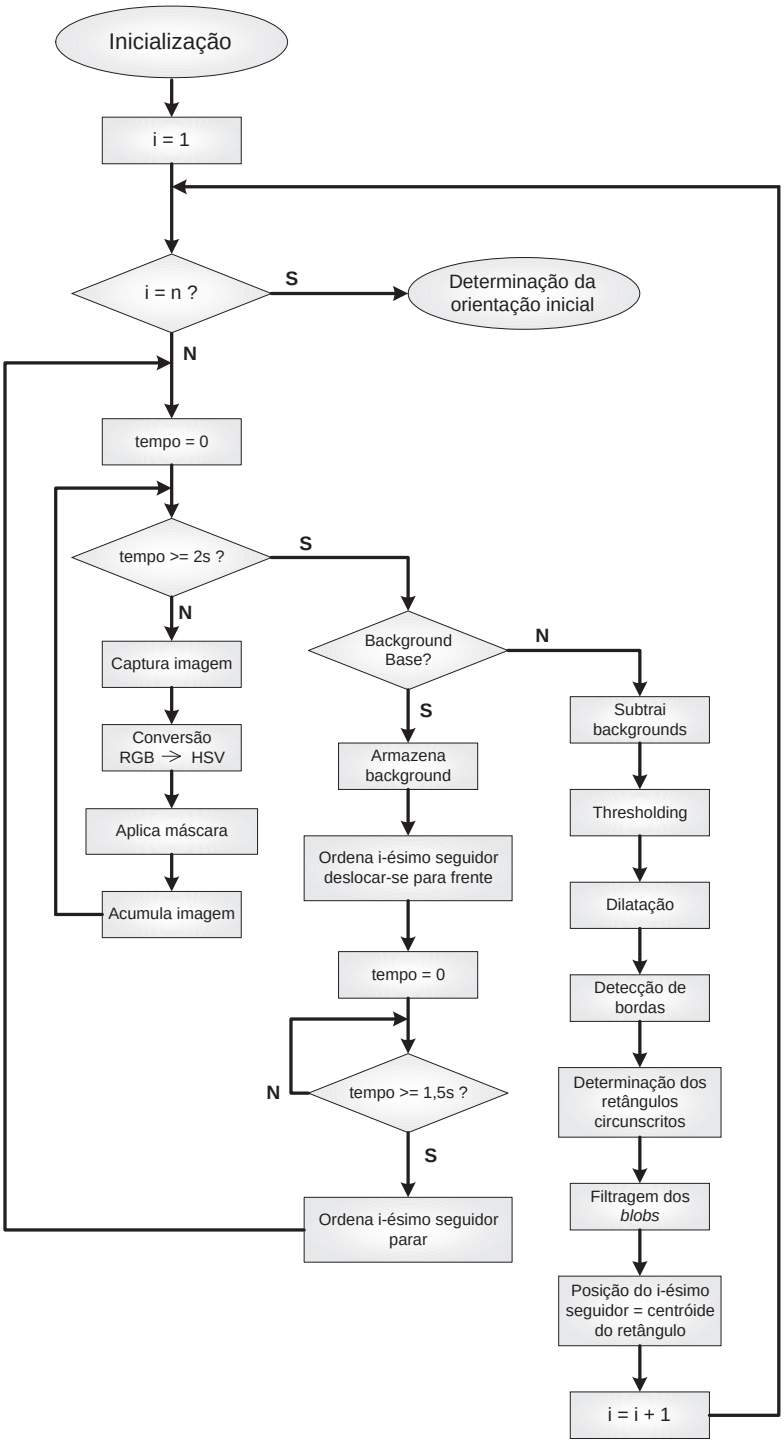


Figura 3.8: Fluxograma simplificado da etapa de detecção das posições iniciais dos seguidores.

Agora, usando a região de interesse definida pelo retângulo circunscrito ao *blob*, um novo histograma é construído para cada seguidor, sendo, em seguida, associado a ele. Vale notar que a partir desse momento cada seguidor possui não apenas uma cor predominante que o caracteriza, mas também todo um histograma, permitindo uma melhor segmentação de cores.

3.2.2 Rastreamento para Determinação da Orientação Inicial

Uma vez conhecidas as posições iniciais dos seguidores, o passo seguinte é estimar suas orientações iniciais. Normalmente, duas cores são usadas nos robôs, e assim a orientação pode ser facilmente determinada tomando-se o centróide de cada região colorida [33, 54], como mostrado na Figura 3.9 - (a). Obviamente, quando duas cores são usadas, cada uma das partes coloridas representa apenas metade da área total que pode ser vista na imagem, Figura 3.9 - (b). Assim, devido à distorção típica das imagens omnidirecionais, que reduz a imagem do seguidor à medida que este se afasta do líder, áreas coloridas pequenas, como as da Figura 3.9 - (a), tornam difíceis os processos de detecção e segmentação de cores. Claramente, isto prejudica as estimativas de postura do robô, caso uma pequena ou mesmo nenhuma área colorida seja percebida pelo líder.

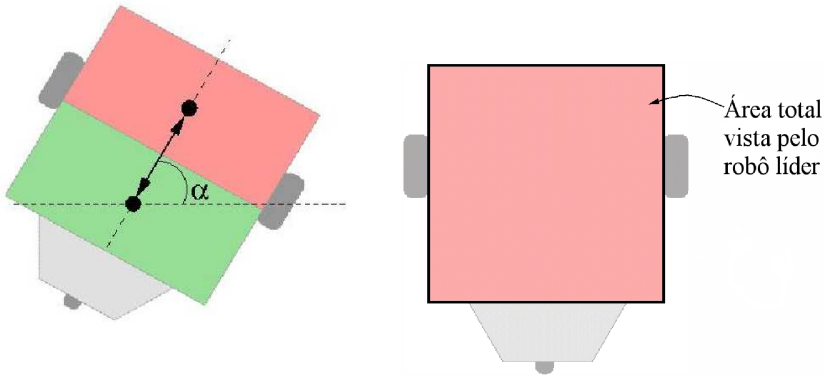


Figura 3.9: (a) Quando duas cores são usadas para determinar a orientação de um robô. (b) Quando apenas uma cor é utilizada, a detecção e segmentação dessa cor é mais simples e mais rápida.

Logo, optou-se pela utilização de apenas uma cor para cada seguidor. Dessa forma, o método usado para determinar a orientação inicial consiste em obter, para cada seguidor, uma seqüência de pontos que descrevem, de forma geral, uma reta. Em seguida, um processo de refinamento de dados é aplicado. Seu objetivo é excluir aqueles pontos que, devido aos erros de medição, prejudicariam a determinação dos parâmetros da reta. Neste trabalho, o refinamento de dados é obtido aplicando-se o algoritmo conhecido como RANSAC (*Random Sample Consensus*) [55]. Trata-se de um algoritmo genérico e muito interessante que identifica, dado um modelo qualquer e um conjunto de pontos, quais os melhores pontos deste conjunto para a determinação dos parâmetros ou coeficientes do modelo fornecido. Todavia, não é sua função fornecer tais parâmetros. Para isso, diferentes técnicas de minimização podem ser usadas. Uma delas é a técnica de Mínimos Quadrados, que foi também adotada neste trabalho. Desta forma,

o modelo fornecido é a equação de uma reta e, dos dois parâmetros que caracterizam esta reta, apenas o coeficiente angular é interessante para esta etapa, pois traduz a orientação do seguidor. Logo, é o único parâmetro a ser minimizado.

Para formar o conjunto de pontos (ou posições) a ser refinado, o líder mais uma vez envia a todos os seguidores comandos de deslocamento para frente e em linha reta. Porém, desta vez, o deslocamento é lento e de forma simultânea, ou seja, os seguidores se movem ao mesmo tempo. O motivo pelo qual os seguidores devem se mover lentamente está no fato de que assim mais pontos serão capturados e, portanto, melhor será a estimativa de orientação. A Figura 3.10 exibe o processo descrito acima para um seguidor.

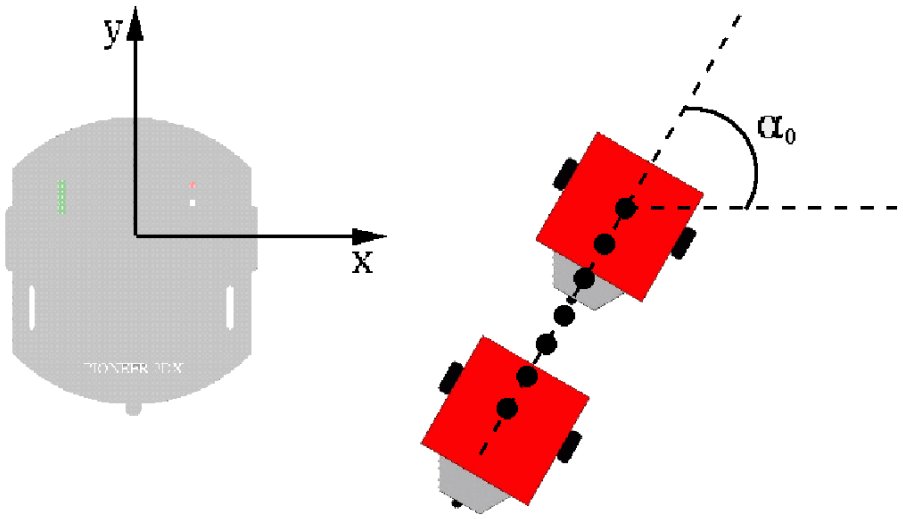


Figura 3.10: Processo de captura das posições de um seguidor para determinação de sua orientação inicial, ou α_0.

A captura das posições de um seguidor continua até que este tenha se deslocado por, pelo menos, 20 cm. Assim que o líder detecta que um determinado seguidor atingiu esta marca, submete o conjunto de pontos deste seguidor ao algoritmo de refinamento. Em seguida, aplica a minimização aos pontos selecionados e obtém uma estimativa da orientação deste seguidor. O líder continua monitorando os outros robôs até que todos tenham sua orientação inicial estimada. A Figura 3.11 mostra um exemplo da aplicação dos algoritmos RANSAC e Mínimos Quadrados a um conjunto de pontos do qual se deseja extrair uma reta.

Como pode ser percebido, à medida em que os seguidores se movem, suas posições vão sendo armazenadas. Para isso, o líder deve ser capar de, a cada imagem capturada, estimar, com boa precisão, a posição de cada seguidor. Portanto, faz-se necessário um algoritmo de rastreamento. Neste momento, entra em ação o principal algoritmo usado no processamento das imagens omnidirecionais deste trabalho. Trata-se de um algoritmo de segmentação por

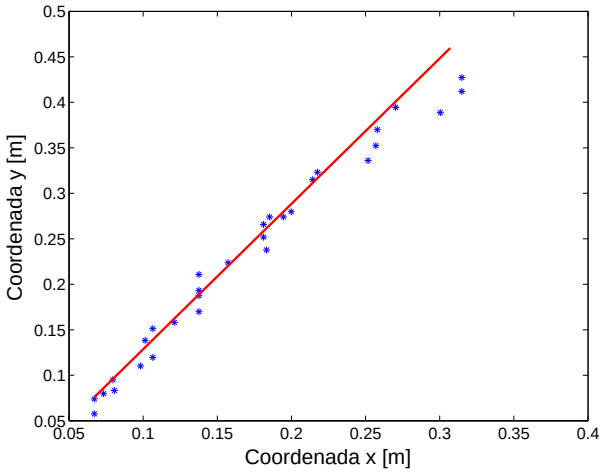


Figura 3.11: Exemplo de aplicação dos algoritmos RANSAC e Mínimos Quadrados.

cores disponível na biblioteca OpenCV e conhecido como CAMSHIFT (*Continuously Adaptive Mean-SHIFT*), que funciona da seguinte forma: o canal *Hue* de cada imagem capturada é usado, juntamente com o histograma da cor de interesse, para gerar uma imagem de distribuição de probabilidade de cor. Daí a importância de associar a cada seguidor o seu histograma de cores, que servirá de modelo para o algoritmo.

Dada uma janela de busca inicial, que é o retângulo circunscrito ao *blob* determinado na etapa anterior, o algoritmo encontra, no interior e nas proximidades desta janela, os *pixels* que apresentam cor suficientemente próxima daquela caracterizada pelo histograma. Esta região é, então, segmentada. O centro e o tamanho dessa região são encontrados e utilizados pelo algoritmo para gerar uma janela de busca inicial na próxima imagem a ser processada. Isto significa que não é necessário varrer toda a imagem para encontrar o objeto de interesse, mas apenas buscar nas proximidades da janela de busca inicial. Isso confere ao rastreamento a rapidez e a robustez necessárias para garantir um bom desempenho do sistema. A Figura 3.12 exibe o algoritmo de rastreamento em ação: os contornos retangulares indicam as regiões coloridas segmentadas que identificam os seguidores. Os histogramas de cada um também podem ser vistos nesta figura.

A Figura 3.13 exibe o fluxograma simplificado do processo de determinação da orientação inicial dos seguidores.

A próxima seção descreve como o líder utiliza o processamento das imagens omnidirecionais para fornecer a realimentação do controlador.

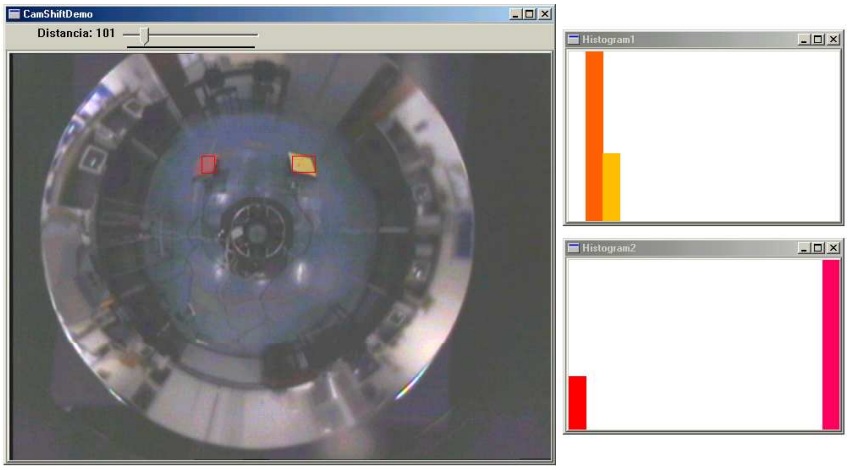


Figura 3.12: Algoritmo de rastreamento em operação e histogramas dos seguidores.

3.2.3 Rastreamento para Controle de Formação

Uma vez conhecidas as posturas iniciais dos seguidores, o passo seguinte seria o líder iniciar a navegação e o controle de formação, enviando para os demais robôs da equipe os comandos que os conduzem à formação desejada. Entretanto, antes disso, a seguinte pergunta deve ser respondida: dada uma posição desejada, qual dos seguidores deve se dirigir a ela? Isso é necessário para evitar que dois ou mais seguidores se dirijam à mesma posição, o que, obviamente, não conduz à formação de interesse.

Deve-se fazer, aqui, uma observação importante: o fato de todos os robôs da equipe serem não-holonômicos impõe a restrição de que, uma vez atingida a formação desejada, a orientação dos seguidores será a mesma do líder. Para isso, a orientação em relação ao líder deve ser (sempre) de 90^o. Por outro lado, a posição desejada varia de acordo com a formação escolhida. Por isso que na pergunta acima usou-se o termo *posição* desejada e não *postura* desejada.

Para determinar qual dos seguidores deve ser direcionado para uma dada posição, define-se a matriz de custo C, como na Equação 3.1.

$$C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & & \ddots & \vdots \\ c_{n1} & c_{n2} & \cdots & c_{nn} \end{bmatrix}, \tag{3.1}$$

onde n é o número de seguidores e c_{ij} representa o custo para o i-ésimo seguidor alcançar a j-ésima posição desejada. Logo, a i-ésima linha da matriz C é o vetor com os custos do i-ésimo seguidor em relação a todas as posições desejadas. Nota-se também que C será sempre uma matriz quadrada, pois para n seguidores devem existir n posições.

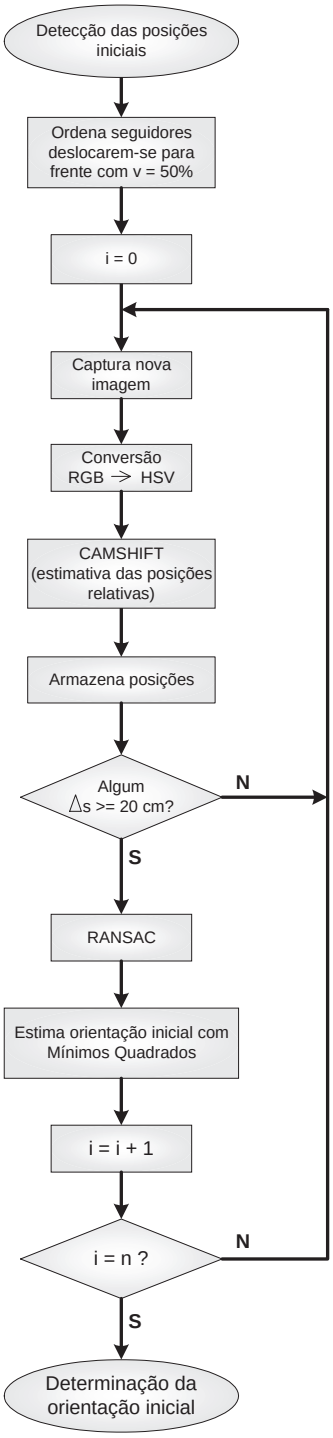


Figura 3.13: Fluxograma simplificado da etapa de detecção das orientações iniciais.

Dentre as maneiras de se definir como calcular os custos c_{ij}, optou-se, neste trabalho, pelo quadrado da distância euclidiana entre as coordenadas do i-ésimo seguidor e da j-ésima posição desejada. O motivo pelo qual se trabalha com o quadrado das distâncias é evitar o cálculo da raiz quadrada, procedimento computacionalmente custoso e totalmente desnecessário neste caso.

O próximo passo consiste em definir uma função que indique qual a melhor configuração, ou seja, dentre todas as possíveis combinações seguidor-posição, qual deve ser adotada. Assim, define-se a função de custo F_c como o somatório de n custos, cada um tomado em uma linha e uma coluna de C diferentes de todos os demais. Cada combinação seguidor-posição possui uma, e apenas uma, função de custo associada. Logo, para n seguidores, existem $n!$ combinações e, portanto, são necessárias $n!$ funções de custo.

Além disso, pode-se aproveitar a analogia com a energia despendida no deslocamento que cada seguidor faria para alcançar uma determinada posição e compreender que, dentre todas as combinações possíveis, a configuração ideal é a que apresenta a menor energia associada.

Assim que forem definidos os pares seguidor-posição desejada, o líder inicia, finalmente, a navegação. Para gerar os comandos necessários ao controle da formação, como já se sabe, é preciso estimar as posturas dos seguidores. Todavia, devido aos ruídos presentes na imagem e a distorção causada pelo espelho curvo, estimar as posturas a cada nova imagem capturada é impraticável. A solução proposta consiste em atualizar a postura do seguidor assim que este apresentar um deslocamento considerado mínimo, reduzindo a sensibilidade aos ruídos.

Assim, o procedimento para estimar a postura de um seguidor é dividido em duas fases. A primeira, estimativa de posição, é obtida diretamente do processamento de uma imagem, enquanto que a segunda, estimativa de orientação, é conseguida de forma indireta, através de um método que utiliza os resultados obtidos na primeira fase.

A estimativa de posição conta com o algoritmo de rastreamento (CAMSHIFT) que fornece, a cada imagem capturada, as posições relativas dos seguidores. Uma maneira de obter a orientação pode ser encontrada em [34, 37]. Entretanto, optou-se pelo desenvolvimento de um método mais simples, de menor custo computacional, baseado na geometria da trajetória descrita pelo seguidor e suas velocidades linear e angular desejadas. Este método é aplicado assim que o seguidor realiza o deslocamento mínimo (Δs_{min}). A posição e a orientação recém estimadas são usadas para atualizar a postura do seguidor.

Portanto, embora a posição seja estimada a cada nova imagem capturada, a postura só é atualizada e repassada ao controlador após o deslocamento mínimo. Com isso, enquanto a postura não é atualizada, os comandos de velocidade linear e angular permanecem constantes, fazendo o seguidor descrever, no caso geral, uma circunferência. A geometria da trajetória, para o caso geral, pode ser vista na Figura 3.14.

Na Figura 3.14, $P_1(x_1, y_1)$ e $P_2(x_2, y_2)$ são, respectivamente, as posições anterior e atual do seguidor. $O(x_0, y_0)$ é o centro da circunferência descrita pelo robô, sendo r o seu raio. Δs

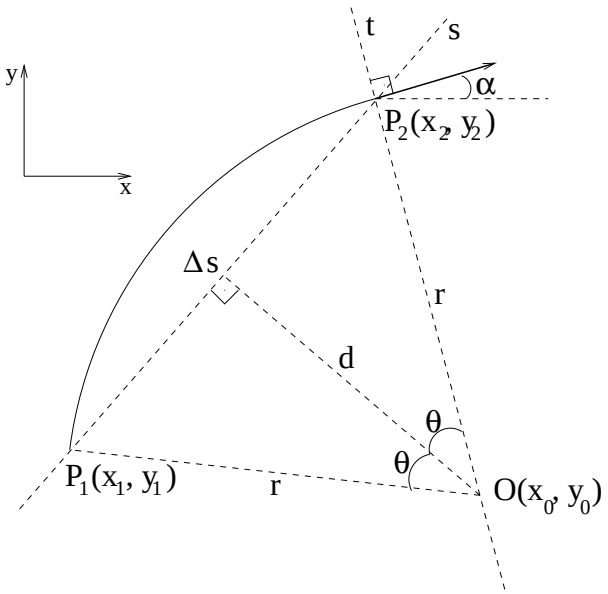


Figura 3.14: Geometria da trajetória descrita pelo seguidor enquanto sua postura não é atualizada.

é a distância entre P_1 e P_2, d é a distância entre O e a reta s que contém P_1 e P_2. t é a reta que passa por P_2 e O, θ é a metade do ângulo $P_1\widehat{O}P_2$ e α é a orientação atual do seguidor. O objetivo é calcular α e, para isso, basta saber a inclinação da reta t. Isto, por sua vez, exige o conhecimento das coordenadas x_0 e y_0 do ponto O. Assim, a distância de O a P_1 é dada pela Equação 3.2.

$$(x_0 - x_1)^2 + (y_0 - y_1)^2 = r^2 \tag{3.2}$$

Da mesma forma, a distância de O a P_2 é dada pela Equação 3.3.

$$(x_0 - x_2)^2 + (y_0 - y_2)^2 = r^2 \tag{3.3}$$

Como O é equidistante de P_1 e P_2, o resultado da manipulação algébrica das Equações 3.2 e 3.3 é dado por:

$$x_0 = Ay_0 - B, \ \ onde \tag{3.4}$$

$$\begin{aligned} A &= \frac{y_1 - y_2}{x_2 - x_1} \quad para \quad x_2 \neq x_1 \\ &\quad e \\ B &= \frac{1}{2}\left[\frac{(y_1 + y_2)(y_1 - y_2)}{x_2 - x_1} - x_1 - x_2\right]. \end{aligned}$$

A reta s pode ser descrita pela Equação 3.5.

$$y = m_s x + l_s, \tag{3.5}$$

onde m_s e l_s são, respectivamente, os coeficientes angular e linear da reta, e podem ser determinados por

$$\begin{aligned} m_s &= \frac{y_2 - y_1}{x_2 - x_1} \\ &e \\ l_s &= y_2 - m_s x_2. \end{aligned}$$

A distância d pode ser determinada observando-se que

$$\tan\theta = \frac{\frac{\Delta s}{2}}{d}.$$

Logo,

$$d = \frac{\Delta s}{2\tan\theta}.$$

O ângulo θ pode ser calculado através da relação

$$2\theta = \omega \times \Delta t,$$

e, portanto,

$$\theta = \frac{\omega \times \Delta t}{2}, \tag{3.6}$$

onde ω é a velocidade angular (constante) do seguidor e Δt é o tempo transcorrido para o seguidor ir de P_1 a P_2. O próximo passo é relacionar x_0 e y_0 à distância d. Para isso, foi utilizada a expressão para calcular a distância entre um ponto e uma reta, obtendo-se

$$d = \frac{|-m_s x_0 + y_0 - l_s|}{\sqrt{m_s^2 + 1}}. \tag{3.7}$$

Substituindo 3.4 em 3.7 e isolando y_0 obtém-se

$$y_0 = \frac{|d|\sqrt{m_s^2 + 1} + l_s - m_s B}{1 - m_s A}. \tag{3.8}$$

Observando que $m_s = -A$, a Equação 3.8 pode ser reescrita como

$$y_0 = \frac{|d|\sqrt{A^2 + 1} + l_s + AB}{1 + A^2}, \tag{3.9}$$

eliminando-se a necessidade de calcular m_s. A inclinação da reta t (m_t) é determinada por

$$m_t = \frac{y_2 - y_0}{x_2 - x_0}.$$

Como a orientação do seguidor é perpendicular à reta t, tem-se que

$$\tan\alpha = \frac{-1}{m_t} \Rightarrow \alpha = \tan^{-1}\left(\frac{x_0 - x_2}{y_2 - y_0}\right).$$

Como as Equações 3.4 e 3.9 fornecem, juntas, dois possíveis pontos, O e O', o procedimento para encontrar O é

1 - Calcular d da seguinte forma: $d = |\frac{\Delta s}{2\tan\theta}|$

2 - Se $(\omega < 0 \quad e \quad x_2 > x_1)$ OU $(\omega > 0 \quad e \quad x_2 < x_1)$, d deve ser considerado negativo na Equação 3.7.

Para o caso em que $x_2 = x_1$, basta verificar se $y_1 > y_2$. Se verdadeiro, $\alpha = \frac{-\pi}{2}$; se falso, $\alpha = \frac{\pi}{2}$.

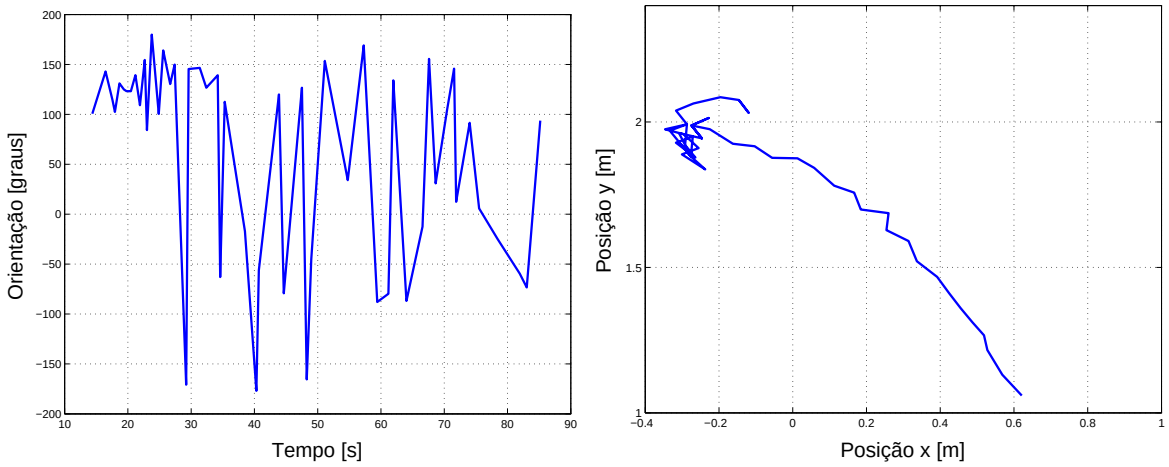
Vale ressaltar que à medida que $\omega \to 0$, $d \to \infty$. Entretanto, esse caso foi tratado na implementação do método, onde o valor de ω é verificado antes de se aplicarem as equações apresentadas acima.

Para validar o método proposto de estimativa de orientação, foram realizados testes com valores de 15, 10 e 5 cm para o deslocamento mínimo. Os melhores resultados foram obtidos com 5 cm e, por isso, este foi o valor adotado para o deslocamento mínimo. Por outro lado, estes testes iniciais serviram também para mostrar a necessidade de se aplicar uma filtragem aos valores estimados de orientação. Em outras palavras, mesmo com a adoção do deslocamento mínimo, a estimativa de orientação ainda é excessivamente ruidosa, como pode ser visto na Figura 3.15-(a). Pode-se notar, também, que até a trajetória descrita pelo seguidor é afetada, tornando-se irregular (Figura 3.15-(b)). Isso acontece porque uma má estimativa da orientação prejudica a estimativa da postura, gerando sinais de comando errôneos.

Este teste foi feito com apenas dois robôs: o líder, que permaneceu parado, e um seguidor. As coordenadas da posição desejada para o seguidor são $x_d = -0.6\ m$ e $y_d = 1.9\ m$.

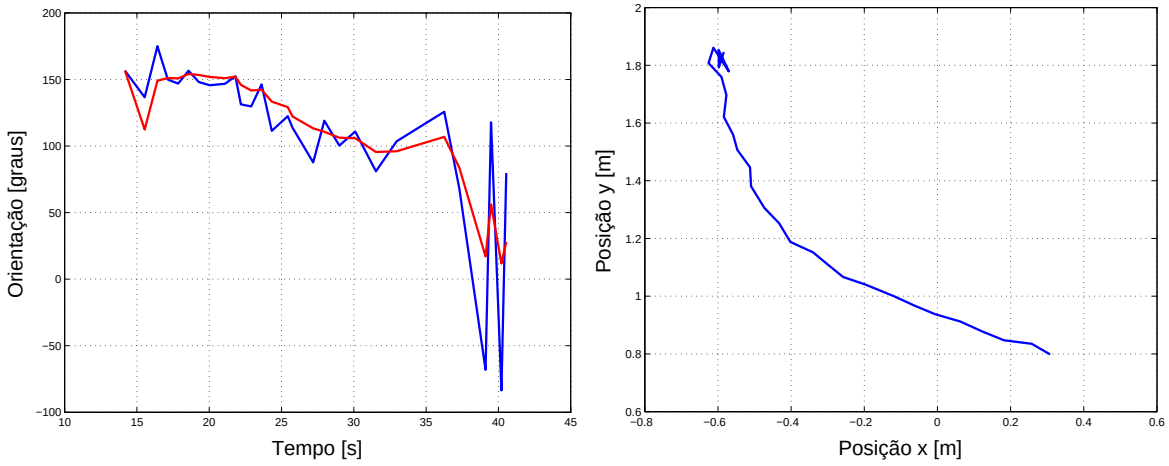
Com o objetivo de reduzir o efeito desses ruídos, alguns filtros foram estudados, como filtros polinomiais, $\alpha - \beta$, Kalman e filtros adaptativos. Dentre eles, foi escolhido o Filtro de Kalman, devido ao seu bom desempenho e relativa simplicidade. Esta última característica tem importante impacto no custo computacional, uma vez que a filtragem é executada a cada atualização de postura.

Assim, novos testes foram feitos utilizando a filtragem da orientação calculada. Os resultados podem ser observados na Figura 3.16. Em azul são exibidos os valores calculados e em vermelho, os valores filtrados. Nota-se que a filtragem melhora significativamente a estimativa da orientação.



(a) (b)

Figura 3.15: Teste sem filtragem. (a) Estimativa da orientação. (b) Trajetória descrita pelo seguidor.



(a) (b)

Figura 3.16: Teste com filtragem. (a) Estimativa da orientação. (b) Trajetória descrita pelo seguidor.

A Figura 3.16-(b) mostra a trajetória descrita pelo seguidor durante este último teste. Comparando-a com a apresentada na Figura 3.15-(b), percebe-se que a filtragem, ao melhorar a estimativa da orientação (e, portanto, da postura), contribui para a geração de comandos mais suaves, resultando em trajetórias mais suaves.

Por fim, é importante esclarecer o que provoca a má estimativa de orientação, mesmo com filtragem, no final dos testes: à medida que o seguidor se aproxima da posição desejada, a sua velocidade linear deve diminuir; ou seja, o comando de velocidade linear é proporcional ao erro de posição. Entretanto, os motores utilizados nos robôs seguidores possuem grande redução, o que lhes confere uma inércia relativamente alta. A velocidade linear de comando recebida

pelo seguidor é convertida em um valor de PWM (*Pulse Width Modulation*) proporcional ao módulo desta velocidade. A máxima velocidade corresponde a um valor de 100% para o PWM. Um valor de PWM abaixo de 20% não é suficiente para acionar os motores devido à inércia. Assim, o robô tende a parar antes de alcançar a posição desejada. Quando o valor do PWM cai abaixo de 30%, o movimento do robô começa a se tornar irregular, não executando as velocidades linear e angular corretamente. Por outro lado, o cálculo da orientação é baseado no conhecimento da velocidade angular de comando ω (Equação 3.6), e não na velocidade angular que está realmente sendo executada, fazendo com que o líder "se engane" a respeito da verdadeira postura do seguidor e gere comandos indesejados. Entretanto, quando toda a equipe estiver navegando, este problema irá desaparecer, pois o robô seguidor deverá estar em movimento.

O fato do sistema de visão estar embarcado no líder tem a grande vantagem de tornar a área de navegação ilimitada, ou seja, o sistema de referência navega junto com a equipe. Porém, para que a estimativa de orientação ocorra, é imprescindível calcular o Δs de cada seguidor. Supondo que um determinado seguidor já se encontre na postura desejada, seu deslocamento na imagem é nulo. Por outro lado, suas velocidades linear e angular não são necessariamente nulas. Fica claro, então, que é preciso um meio de obter o Δs de cada seguidor com o líder em movimento. Para determinar o deslocamento de um robô deve-se conhecer as suas posições anterior e atual. Como o sistema de coordenadas freqüentemente sofre translação e rotação, uma transformação precisa ser aplicada à última postura, para que esta seja corretamente representada no novo sistema de referência. Desta forma, é possível determinar o Δs de cada seguidor.

A transformação a ser aplicada à *posição* de um seguidor é nada mais que uma transformação geométrica 2D. Como o líder possui, de um modo geral, velocidades angular e linear, ela é a composição de uma rotação e de uma translação, conforme ilustra a Figura 3.17.

De acordo com a Figura 3.17-(c), γ é o ângulo de rotação, e dx e dy são as translações em x e y, respectivamente. As coordenadas de P_0 no sistema S_0 são $(x_0 \ y_0)^T$, como na Figura 3.17-(a). Para o sistema S_1 (Figura 3.17-(b)), este ponto, agora chamado de P_1, possui coordenadas $(x_1 \ y_1)^T$. Assim, para obter P_1 conhecendo P_0, basta aplicar a transformação mostrada na Equação 3.10.

$$P_1 = R(-\gamma)\,T(-dx,\ -dy)\,P_0 \tag{3.10}$$

A Equação 3.10 pode ser reescrita em sua forma matricial, como na Equação 3.11.

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} cos\gamma & -sen\gamma & 0 \\ sen\gamma & cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -dx \\ 0 & 1 & -dy \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \tag{3.11}$$

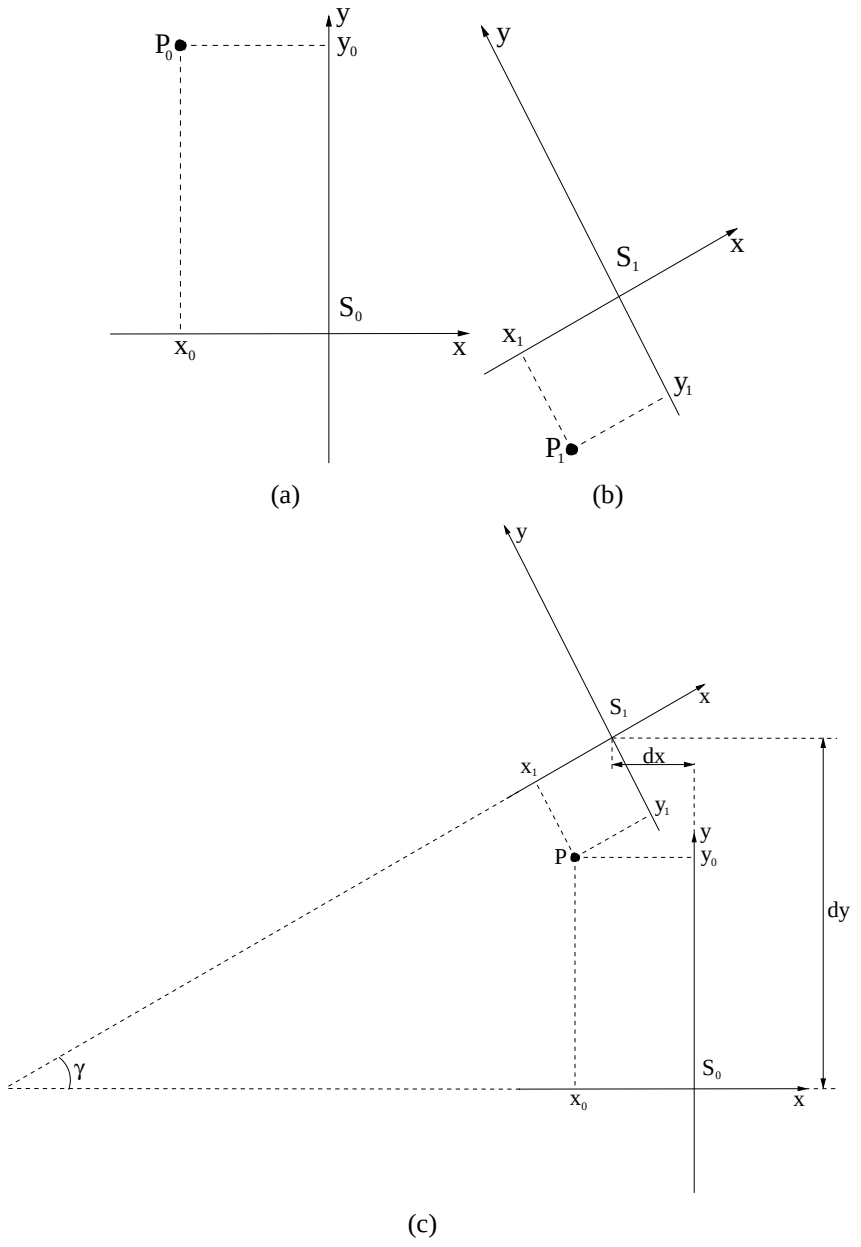


Figura 3.17: Efeito da composição de uma rotação e de uma translação de um sistema de referência nas coordenadas de um ponto.

Assim, realizando as operações mostradas na Equação 3.11, é possível conhecer a posição anterior de um seguidor no novo sistema de coordenadas e, portanto, calcular o respectivo deslocamento. Devido à geometria de projeção de um sistema omnidirecional, uma translação deste sistema não provoca alteração na orientação dos seguidores. Entretanto, uma rotação causa esse problema em toda a equipe. Este efeito está ilustrado na Figura 3.18.

Uma análise simples da Figura 3.18 mostra que após uma rotação do líder de um ângulo γ, a nova orientação relativa de um seguidor, α_1, será dada por $\alpha_1 = \alpha_0 - \gamma$.

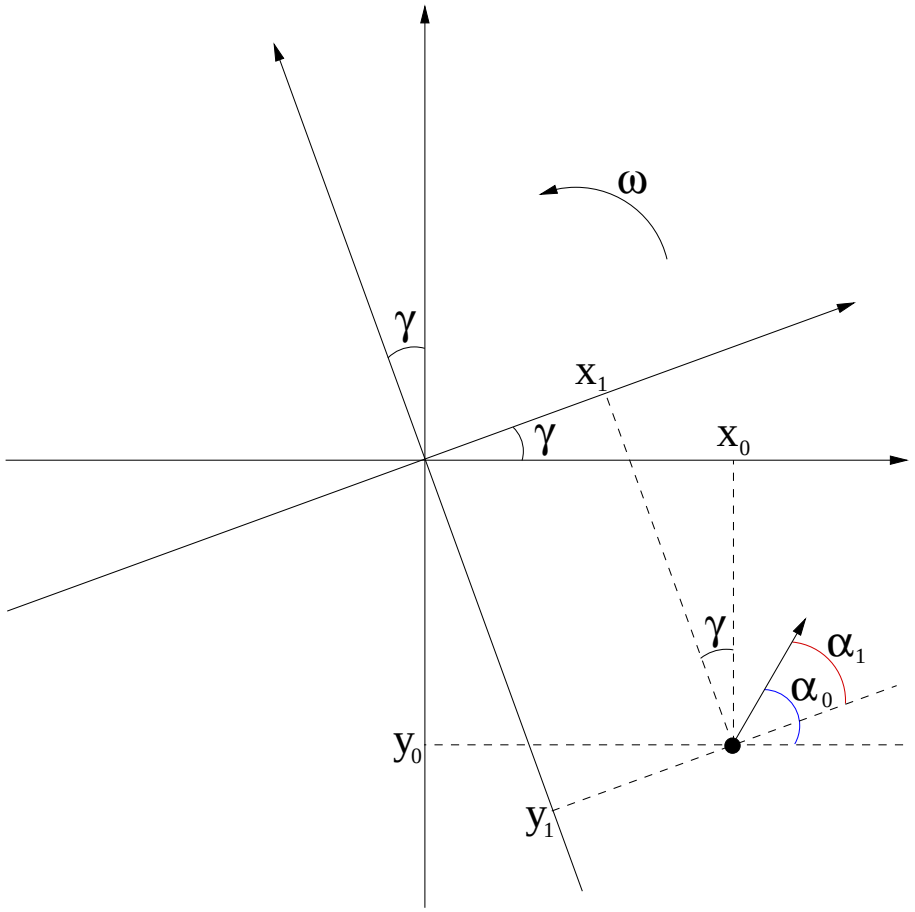


Figura 3.18: Efeito da rotação do robô líder na orientação relativa de um seguidor.

Assim, foi definida uma transformação Λ responsável pela atualização da postura, ou seja, atualiza tanto a posição quanto a orientação dos seguidores. Para isso, bastou uma pequena alteração na Equação 3.11 que, após a manipulação algébrica, resulta na Equação 3.12. Para tornar possíveis as operações matriciais, as posturas no novo e no antigo sistema de coordenadas, χ_1 e χ_0, foram representadas em coodenadas homogêneas por χ_1' e χ_0'.

$$\begin{aligned} \chi_1' &= \Lambda\left(\chi_0', \gamma, dx, dy\right) \\ \begin{bmatrix} x_1 \\ y_1 \\ 1 \\ \alpha_1 \end{bmatrix} &= \begin{bmatrix} cos\gamma & -sen\gamma & -dxcos\gamma - dysen\gamma & 0 \\ sen\gamma & cos\gamma & dxsen\gamma - dycos\gamma & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \\ \alpha_0 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 0 \\ \gamma \end{bmatrix} \end{aligned} \tag{3.12}$$

Desta forma, esta etapa é responsável por estimar as posturas dos seguidores mesmo com toda a equipe em movimento, sendo, portanto, a etapa mais importante do processamento de imagens. A Figura 3.19 apresenta o fluxograma simplificado relativo a esta etapa.

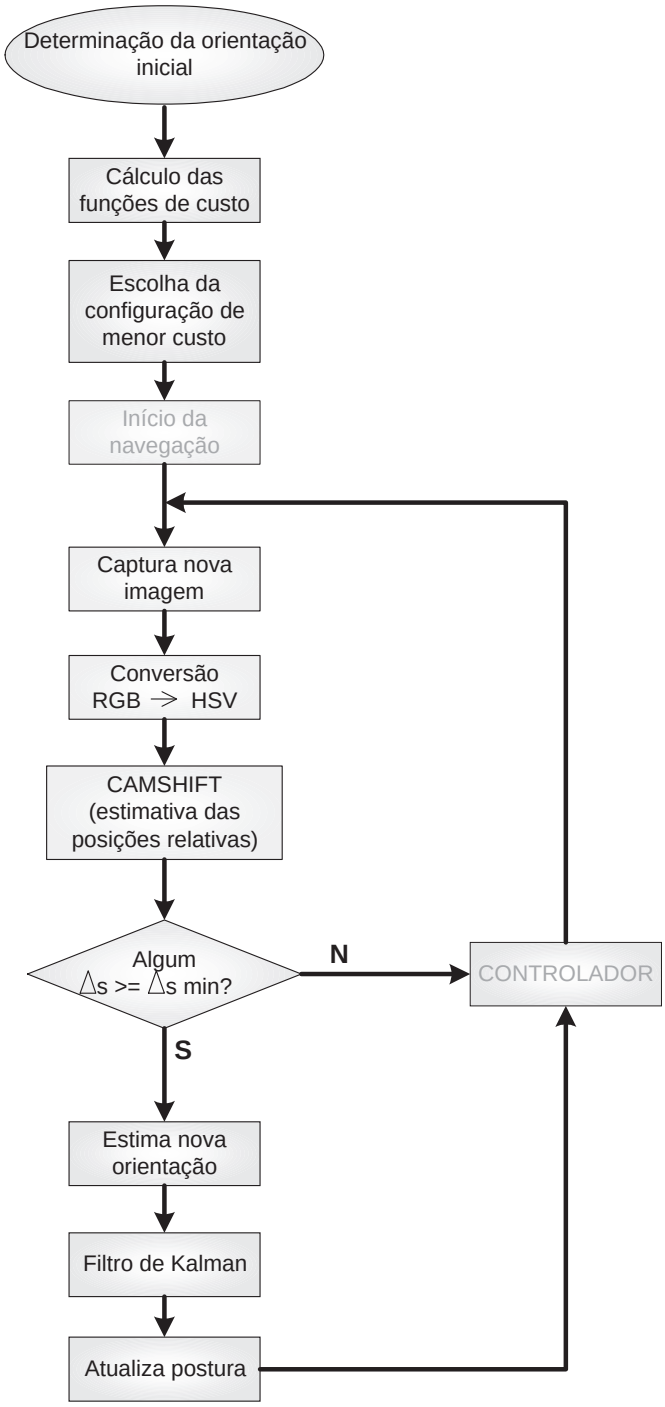


Figura 3.19: Fluxograma simplificado da etapa de Rastreamento para Controle de Formação.

3.3 Conclusões

Este capítulo foi dedicado ao processamento das imagens omnidirecionais, cuja função é servir de realimentação para o controlador empregado. Para isso, foram utilizadas técnicas largamente conhecidas na literatura, como *background subtraction*, dilatação, operações morfológicas e segmentação por cores, além da biblioteca de processamento de imagens OpenCV.

Ainda, novos algoritmos foram desenvolvidos, com o objetivo de aumentar a precisão e a robustez das etapas envolvidas no referido processamento, tais como os algoritmos de determinação das posições iniciais, filtragem dos *blobs* e atualização das posturas dos seguidores com a equipe navegando.

Além disso, técnicas de filtragem de dados, como os algoritmos RANSAC e Filtro de Kalman também foram empregadas, contribuindo de forma decisiva para um bom desempenho do processamento das imagens omnidirecionais.

No próximo capítulo, serão mostrados e discutidos os experimentos realizados para validar o controlador não linear. Também será possível avaliar o comportamento deste controlador mediante os problemas encontrados, como os já citados ruídos presentes na imagem, a distorção causada pelo espelho e a má qualidade das leituras dos *encoders* do robô líder.

4 Resultados Experimentais

4.1 Introdução

Para validar o controlador proposto, bem como os algoritmos de processamento de imagens empregados, foram realizados experimentos com uma equipe formada por três robôs. O robô com maior capacidade de processamento foi escolhido como líder do grupo, enquanto os outros, bem mais simples e baratos, ficaram como seguidores.

O robô utilizado como líder é o PIONEER 2-DX, equipado com um processador PENTIUM MMX, de 233 MHz, 128 MB de memória RAM e fabricado pela ActivMedia Robotics. Além de *encoders* nas rodas, este robô possui ainda uma câmara CCD Sony PTZ (*Pan, Tilt, Zoom*) e oito sensores de ultra-som, como pode ser visto na Figura 4.1. Mais informações sobre o PIONEER 2-DX podem ser encontradas em [56].

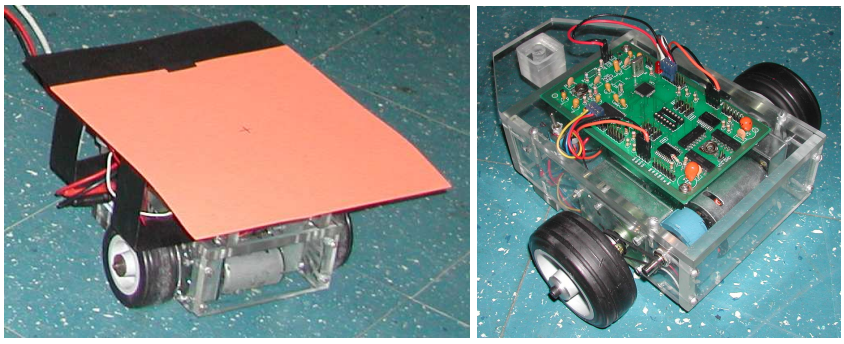
Figura 4.1: Robô PIONEER 2-DX usado como líder da equipe.

Neste trabalho, os *encoders* foram utilizados apenas para a geração das figuras com as trajetórias descritas pela equipe. A câmara Sony não foi usada, uma vez que o sistema de visão foi construído com uma câmara modelo CCS-212 da Samsung e um espelho hiperbólico. Este sistema já foi mostrado na Figura 2.2 e é, por conveniência, exibido também na Figura 4.2.

Uma das motivações deste projeto é a construção de uma equipe de robôs de baixo custo para a realização de tarefas em cooperação. Estes foram construídos no Laboratório de Automação Inteligente (LAI), ligado ao Departamento de Engenharia Elétrica da Universidade Federal

Figura 4.2: Sistema omnidirecional montado sobre o robô líder.

do Espírito Santo – UFES. A Figura 4.3 exibe um destes robôs. A Figura 4.3-(a) mostra, em mais detalhes, o cartão colorido usado pelo algoritmo de segmentação de cores para estimar a posição de cada seguidor. Este cartão é sustentado por uma estrutura de alumínio que serve também para proteger o sistema embarcado (Figura 4.3-(b)). Trata-se de um sistema simples, cuja finalidade é, através de um *link* de comunicação com o líder, receber os comandos de velocidades linear e angular desejadas e aplicar aos motores sinais de PWM para a execução destas velocidades.



(a) (b)

Figura 4.3: Robô seguidor - (a) Cartão colorido usado para estimar sua posição e (b) *Hardware* embarcado.

Este sistema roda no microcontrolador MSP430F149 da Texas Instruments [57, 58], que possui 2 kB de memória RAM e 60 kB de memória FLASH. Além disso, sua CPU de 16 bits otimizada para o uso de linguagens de programação como C, sua arquitetura RISC [1] e o consumo extremamente baixo de energia fazem deste dispositivo uma ferramenta muito interessante em aplicações de robótica móvel. A Figura 4.4 exibe o *hardware* que comporta o microcontrolador e toda a eletrônica necessária à comunicação e acionamento dos motores. Ele foi desenvolvido em [59, 60], onde encontram-se a descrição funcional e os diagramas esquemáticos dos módulos que o compõem.

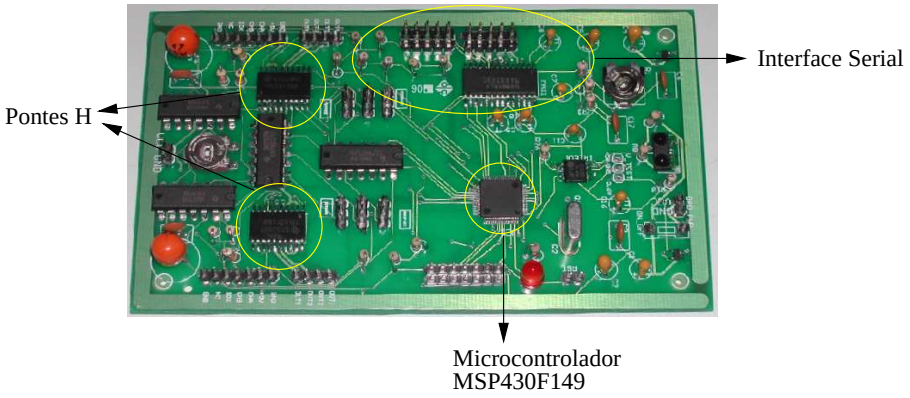


Figura 4.4: *Hardware* embarcado nos robôs seguidores e responsável pelo acionamento dos motores.

4.2 Simulações

Antes de dar início aos experimentos com os robôs, simulações foram realizadas com o intuito de observar o comportamento do controlador mediante as várias formações e trajetórias que a equipe pode descrever. Para isso, através da linguagem de programação C, foi desenvolvido um simulador. Como não se dispõe do modelo dinâmico de cada robô seguidor, este simulador considera apenas os efeitos do controlador na cinemática dos seguidores. Trata-se, portanto, de um simulador simples, mas útil para um bom entendimento do controlador.

Várias simulações foram realizadas. Entretanto, serão mostradas e discutidas três delas, por representarem as trajetórias mais comuns para navegação. São elas: trajetória retilínea, circular e desvio de obstáculo. O passo das simulações foi de 380 ms, que é o tempo médio

[1] RISC significa *Reduced Instruction Set Computer*, ou Computador com Conjunto de Instruções Reduzido. Neste tipo de arquitetura, a CPU gasta apenas um ciclo de máquina para instruções que não provocam desvio na execução do código, e dois ciclos para aquelas que provocam. Assim, mesmo programando em uma linguagem de alto nível, como C ou C++, tendo acesso ao código de montagem gerado (código *Assembly*), é possível estimar em quanto tempo uma determinada rotina é executada. Isso é fundamental para sistemas de tempo real, que possuem grande aplicação no campo da robótica móvel.

para a execução de um loop de controle, e os ganhos usados são os mesmos dos experimentos realizados.

O resultado da primeira simulação é exibido na Figura 4.5. Como pode ser visto, a trajetória descrita é retilínea. A velocidade escolhida para o líder foi de 80 mm/s e esta simulação durou 100 segundos. A linha sólida central indica a trajetória descrita pelo líder, enquanto as outras linhas sólidas de cor azul representam as trajetórias "reais" dos seguidores. As linhas tracejadas mostram as trajetórias desejadas para os seguidores. Por fim, os triângulos vermelhos descrevem como a equipe entra e mantém a formação desejada.

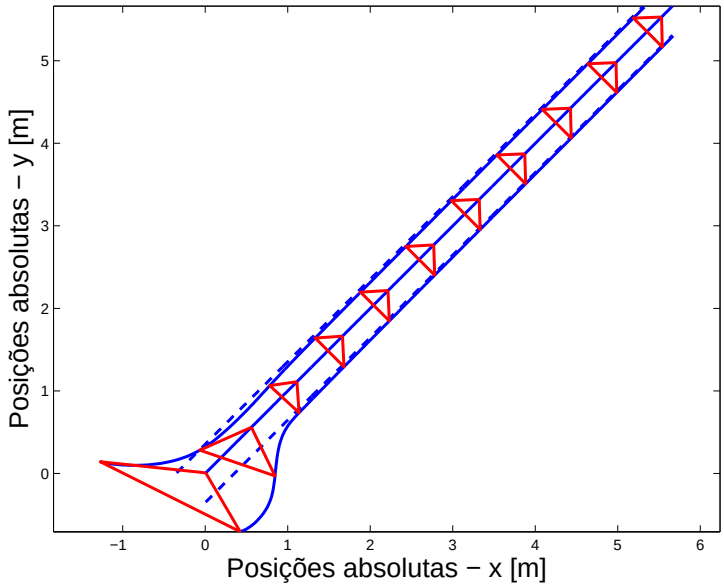


Figura 4.5: Primeira simulação: trajetória retilínea com velocidade de 80 mm/s.

Nesta primeira simulação, as posições iniciais dos seguidores foram, em metros, $x_{01} = -1,0$, $y_{01} = -0,8$, $x_{02} = 0,8$ e $y_{02} = -0,2$ sempre em relação ao líder. As posições desejadas eram $x_{d1} = -0,25$, $y_{d1} = -0,25$, $x_{d2} = 0,25$ e $y_{d2} = -0,25$, ou seja, a equipe forma um triângulo isóceles, como mostra a Figura 4.5. A orientação do líder foi de 45^o em relação ao referencial absoluto, enquanto as orientações dos seguidores foram de 30^o e 45^o em relação ao líder. Nota-se que a equipe entra em formação, embora um pequeno erro na coordenada x possa ser observado. Isso acontece porque a função de saturação relativa à geração da velocidade angular de referência, $f_{\tilde{\alpha}}(\tilde{\alpha}_i)$, tende a zero à medida que os erros tendem a zero. Assim, o seguidor possui uma velocidade angular que é reduzida à medida que este se alinha com o líder. Logo, sua capacidade de girar fica afetada e o erro em x se aproxima de zero de forma mais lenta. Isso não quer dizer que os seguidores não entrarão em formação mas, como ficará claro mais adiante, significa que o simulador, embora simples, retrata bem o comportamento do controlador. Tudo isso pode ser visto na Figura 4.6. A Figura 4.6 - (a) mostra os erros de

posição para o seguidor 1, onde o erro em x é representado pela linha azul e o erro em y pela linha vermelha. A Figura 4.6 - (b) mostra os mesmos erros para o seguidor 2. A Figura 4.6 - (c) traz a evolução das orientações dos seguidores, azul para o seguidor 1 e vermelho para o seguidor 2.

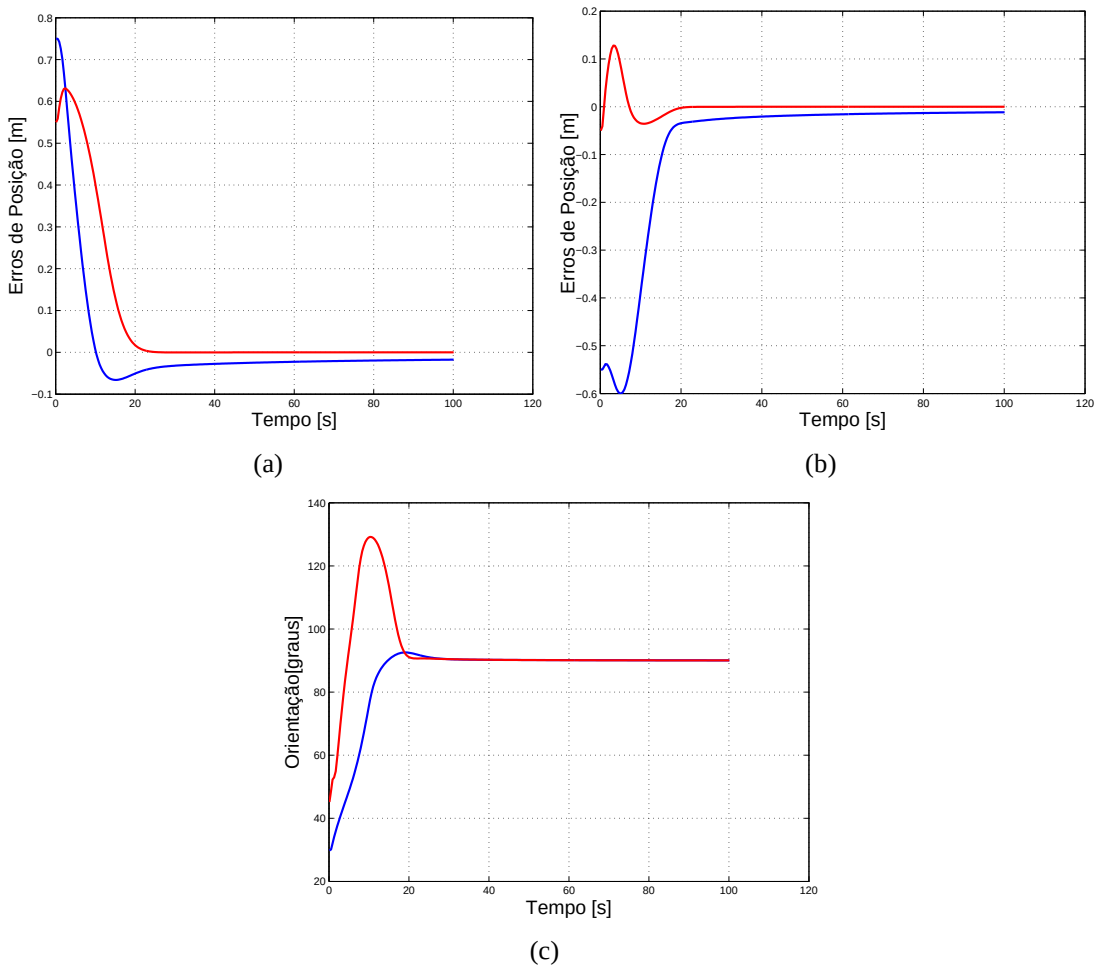


Figura 4.6: Primeira simulação: (a) Erros de posição do seguidor 1, (b) Erros de posição do seguidor 2 e (c) Orientações dos seguidores.

A segunda simulação objetivou verificar o comportamento do controlador para trajetórias circulares. Neste caso, as velocidades linear e angular do líder foram 60 mm/s e 1,5 ^{o}/s, respectivamente. A Figura 4.7 traz o resultado desta segunda simulação para a trajetória. As posições iniciais foram $x_{01} = -0,7$, $y_{01} = 0,0$, $x_{02} = 0,5$ e $y_{02} = -0,9$, enquanto as desejadas eram $x_{d1} = 0,0$, $y_{d1} = 0,7$, $x_{d2} = 0,0$ e $y_{d2} = -0,7$, ou seja, a equipe deveria navegar em linha, com um seguidor a frente do líder e outro atrás. A orientação inicial do líder foi de 0^o no referencial absoluto, enquanto os seguidores partiram com orientações de 30^o e 90^o.

O triângulo vermelho torna-se praticamente uma linha reta, mostrando que a equipe entra e mantém a formação desejada. Esta simulação durou 245 segundos, o suficiente para que o grupo descrevesse uma circunferência completa.

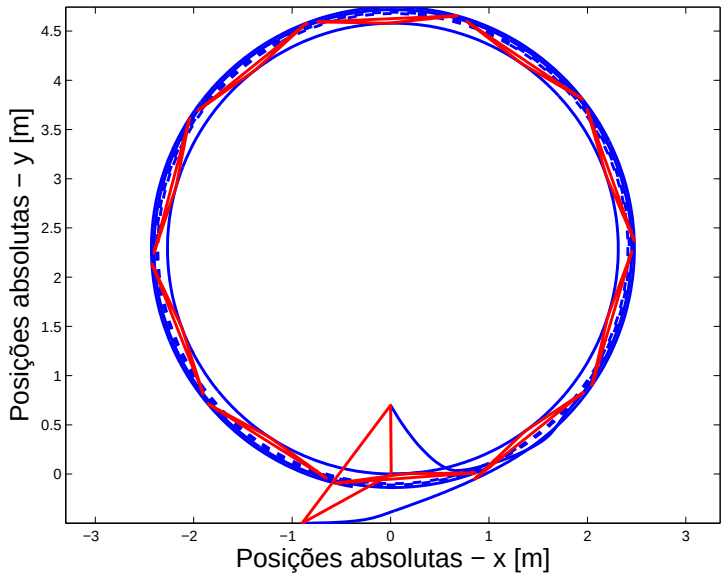


Figura 4.7: Segunda simulação: trajetória circular com velocidade linear de 60 mm/s e angular de 1,5 ^{o}/s.

Nota-se, ainda na Figura 4.7, que o líder descreve um círculo de raio menor que aquele descrito pelos seguidores. Isso já era esperado devido à geometria tanto da formação escolhida quanto da trajetória curva e fica claro observando-se a Figura 4.8 que mostra que, no caso desta simulação, como os seguidores devem permanecer sobre o eixo y do referencial do líder ($x_{d1} = 0,0$ e $x_{d2} = 0,0$), tem-se $r_i = \sqrt{r^2 + y_i^2}$ e, portanto, os raios que eles descrevem são maiores que o do líder. A Figura 4.9 traz os resultados para os erros de posição e para as orientações dos

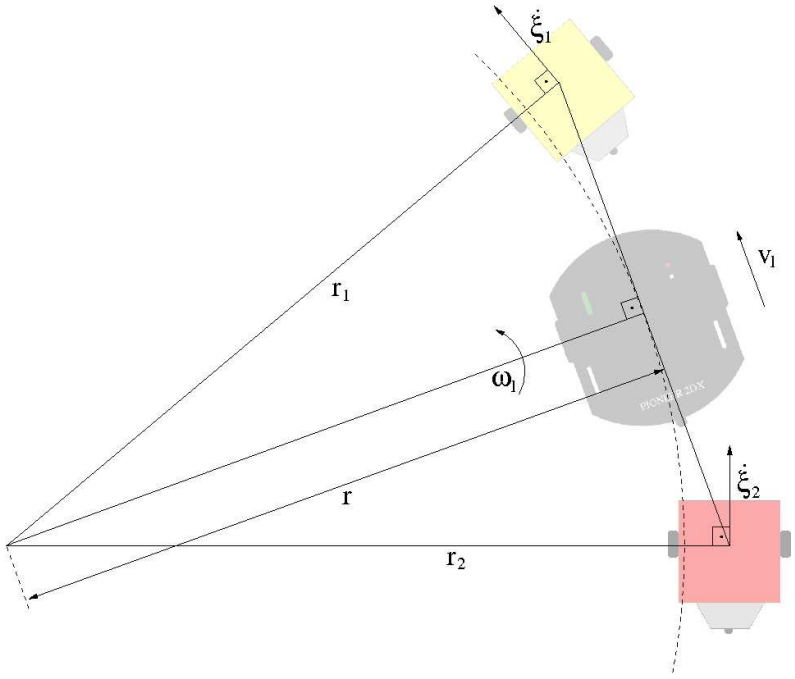


Figura 4.8: Geometria da formação desejada para a segunda simulação.

seguidores nesta segunda simulação. Comparando com os encontrados na primeira, percebe-se que os erros de posição são, agora, maiores. Isso ocorre devido tanto ao movimento de rotação do líder quanto ao passo (380 ms) do simulador. O primeiro efeito pode ser visto da seguinte forma: quando o líder gira, as posições desejadas absolutas giram no sentido contrário, inserindo erros em ambas as coordenadas dos seguidores no próximo passo de simulação. Isso afeta também a orientação de referência: quanto maior o passo da simulação, maiores serão os deslocamentos linear e angular do líder e, portanto, maiores serão os erros inseridos.

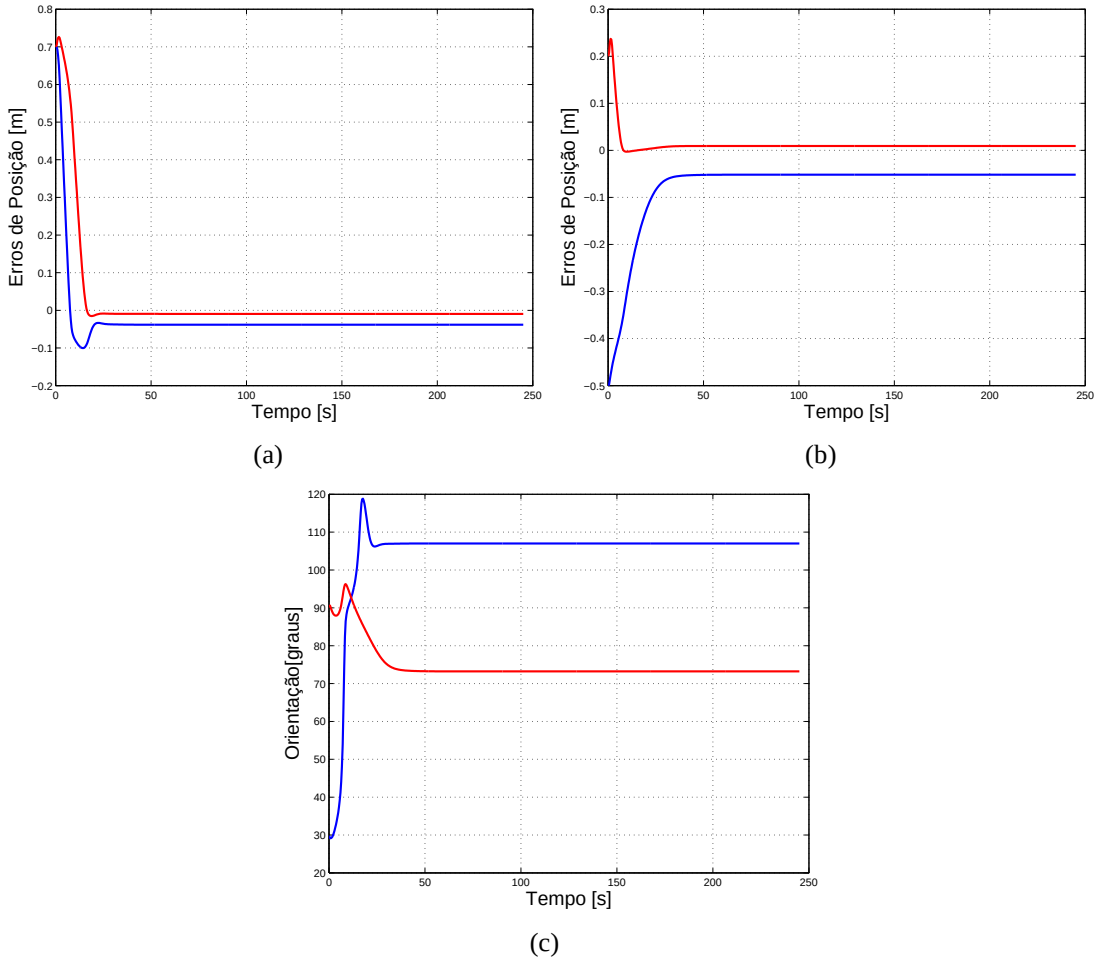


Figura 4.9: Segunda simulação: (a) Erros de posição do seguidor 1, (b) Erros de posição do seguidor 2 e (c) Orientações dos seguidores.

Como mostra a Figura 4.9 - (c), as orientações dos seguidores não estabilizam, desta vez, em 90^o. O motivo é que, para a trajetória ser circular, o seguidor que se encontra a frente do líder deve ter orientação um pouco maior que 90^o. Pela mesma razão, o seguidor que vai atrás precisa ter um pouco menos de 90^o em relação ao líder. Isso é justamente o que mostram as Figuras 4.8 e 4.9 - (c).

A terceira e última simulação aqui mostrada tratou de verificar o comportamento do controlador para uma trajetória sinuosa, ou seja, onde há variação da velocidade angular, inclusive com mudança de sinal. Essa trajetória aparece na Figura 4.10 e, como pode ser visto, simula a situação onde a equipe deve desviar de um obstáculo sem perder a formação. Esta simulação mostra que, a princípio, o controlador apresentará um bom desempenho quando um algoritmo de desvio for acrescentado. Na verdade, a equipe pode desviar de um certo obstáculo ao mesmo tempo que entra em formação. Esta simulação durou 150 segundos e os seguidores estavam inicialmente nas posições $x_{01} = -0,7$, $y_{01} = -0,8$, $x_{02} = 0,3$ e $y_{02} = -0,9$. As posições desejadas eram $x_{d1} = -0,5$, $y_{d1} = -0,3$, $x_{d2} = 0,5$ e $y_{d2} = -0,3$, mais uma vez um triângulo isóceles. A orientação inicial do líder foi de 0^o, enquanto os seguidores partiram com 135^o e 120^o.

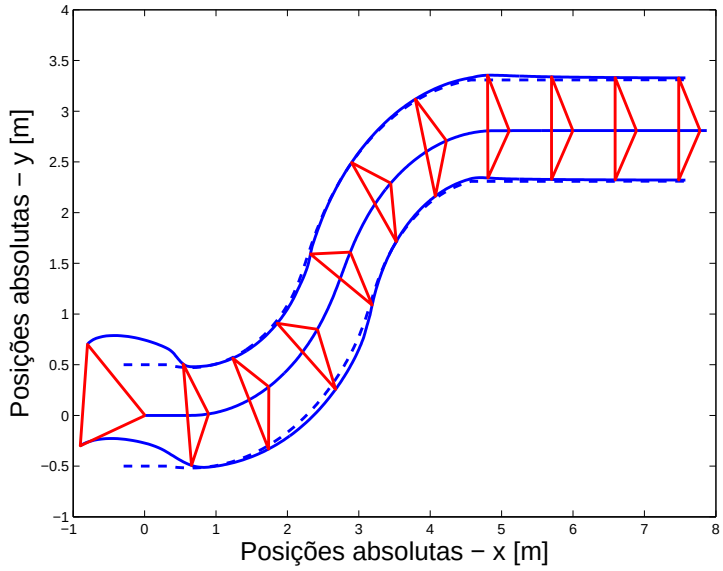


Figura 4.10: Terceira simulação: trajetória sinuosa com velocidade linear de 60 mm/s.

Neste caso, entre os instantes $t_1 = 10s$ e $t_2 = 100s$, a velocidade angular do líder variou obedecendo a uma função do tipo tangente hiperbólica, passando por zero em $t = 55s$. Os valores de saturação foram -1,5 ^{o}/s e +1,5 ^{o}/s. Esta função é apresentada na Equação 4.1, onde $0,0262$ rad/s = 1,5 ^{o}/s.

$$f_{\omega l}(t) = 0,0262 tanh(0,2(55-t)) \tag{4.1}$$

A função $f_{\omega l}(t)$ é a mesma utilizada nos Experimentos 4 e 5, que serão apresentados e discutidos mais adiante. Para os intervalos $0 \leq t < 10s$ e $100 < t \leq 150s$, a velocidade angular foi zero. Nota-se, na Figura 4.11, que é justamente logo após os instantes $t_1 = 10s$, $t = 55s$ e $t_2 = 100s$ que perturbações são causadas no sistema. Isso mostra que a troca de sinal ou mesmo uma variação brusca da velocidade angular tem um efeito muito similar ao de uma perturbação no sistema. Obviamente, a situação proposta nesta última simulação pode ocorrer

com freqüência e o sistema de controle deve ser capaz de manter a estabilidade nestas condições. E é isto que mostram as Figuras 4.10 e 4.11. Portanto, embora os erros apresentados sejam um pouco maiores que nas simulações anteriores, pode-se dizer que a equipe entra e permanece em formação, mesmo sofrendo perturbações.

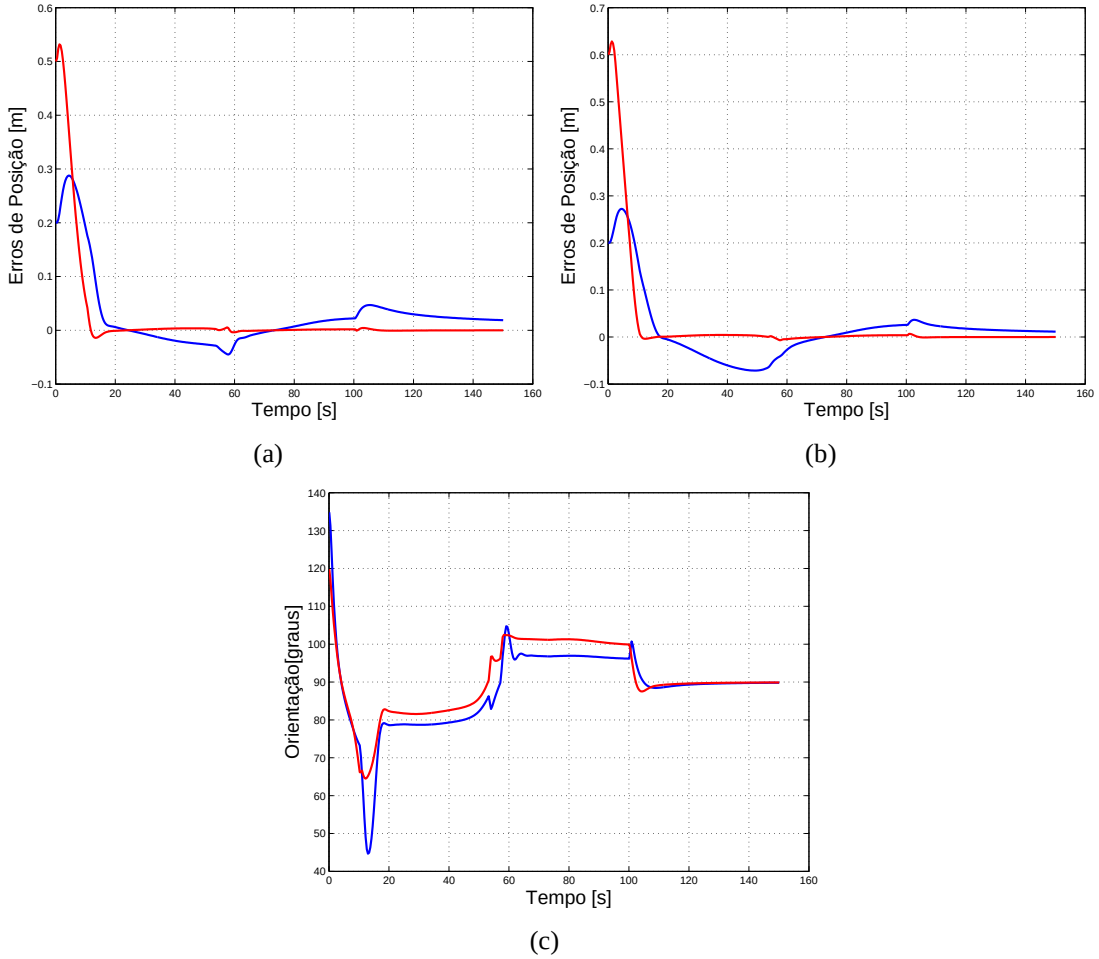


Figura 4.11: Terceira simulação: (a) Erros de posição do seguidor 1, (b) Erros de posição do seguidor 2 e (c) Orientações dos seguidores.

4.3 Os Experimentos

Dentre os vários experimentos realizados, serão mostrados aqui aqueles que apresentaram resultados que permitem uma melhor discussão do comportamento do controlador. É importante dizer que as posições e orientações iniciais apresentadas nos experimentos a seguir correspondem àquelas determinadas após as duas primeiras etapas do processamento de imagens. Além disso, assim como nas simulações, nas figuras que apresentam os erros de posição, a linha azul representa o erro na coordenada x, enquanto o erro na coordenada y é mostrado pela linha vermelha. Já nas figuras que mostram a evolução das orientações dos seguidores, a linha azul

representa a orientação calculada com o algoritmo aqui desenvolvido e a linha vermelha o valor filtrado desta orientação com o Filtro de Kalman. Ainda, também como nas simulações, todas as coordenadas de posição e os valores dos erros são dados em metros.

Nos primeiros experimentos, cada seguidor era identificado por uma cor diferente, como mostrado na Figura 3.12. Os objetivos destes primeiros experimentos foram:

1. Determinar os melhores ganhos para o controlador, ou seja, refinar o controle.
2. Encontrar as cores que resultam em uma melhor segmentação de cores, melhorando a estimativa de postura.

Embora não exista restrição teórica quanto às cores que podem ser usadas para identificar os seguidores, a cor laranja apresentou os melhores resultados. Assim, ela foi utilizada em ambos os seguidores em todos os experimentos subseqüentes. Como a detecção da cor do seguidor é feita de maneira automática, nenhuma alteração no código foi necessária. Entretanto, como a mesma cor foi usada nos dois seguidores, tomou-se o cuidado de evitar que um seguidor passasse muito próximo do outro, pois isto confundiria o algoritmo de segmentação de cores.

Como foi dito no início desta seção, serão apresentados e discutidos os experimentos cujos resultados permitem uma melhor discussão do comportamento do controlador. Estes estão numerados de 1 a 5 e serão detalhados a seguir.

4.3.1 O Experimento 1

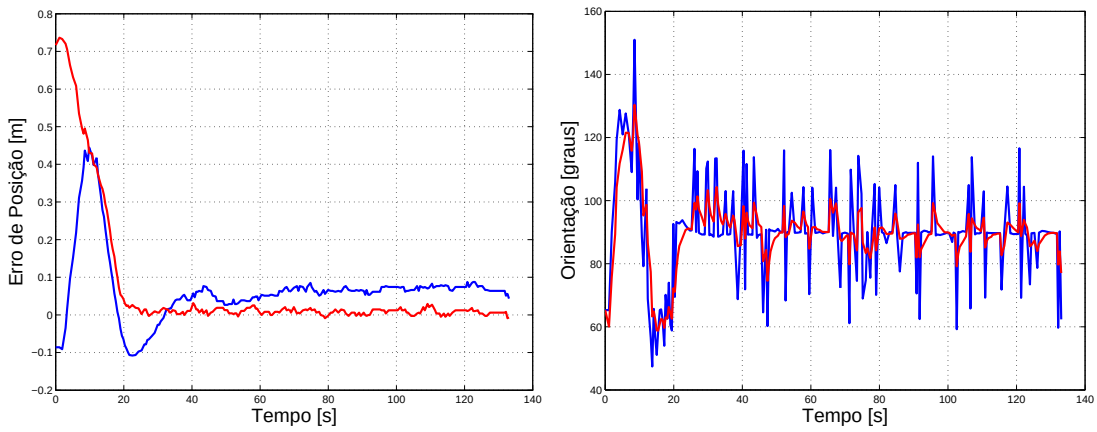
Neste primeiro experimento, o líder desenvolveu uma trajetória retilínea com velocidade constante e igual a 60 mm/s. As posições iniciais foram estimadas em $x_{01} = -0,5$, $y_{01} = -0,4$, $x_{02} = 0,35$ e $y_{02} = -0,5$. As posições desejadas escolhidas foram $x_{d1} = -0,6$, $y_{d1} = 0,3$, $x_{d2} = 0,6$ e $y_{d2} = 0,3$, ou seja, um triângulo isóceles, mas com os seguidores à frente do líder. As orientações iniciais dos seguidores em relação ao líder eram de aproximadamente 135^o para o seguidor 1 e 60^o para o seguidor 2. A duração deste experimento foi de aproximadamente 150 segundos.

A Figura 4.12 mostra os resultados deste experimento para o seguidor 1. Nota-se que, após a oscilação inicial, o erro em x fica limitado a 10 cm. Esta oscilação ocorre principalmente devido ao deslocamento mínimo necessário à atualização da postura, que se traduz como um atraso inserido no controlador. Em outras palavras, até que a postura seja atualizada, os mesmos comandos de velocidade continuam sendo gerados. Quando a postura finalmente é atualizada,

pode ser tarde para evitar que o seguidor passe pela posição desejada, uma vez que os robôs possuem inércia e não respondem instantaneamente aos comandos de velocidade de referência.

A semelhança observada entre os gráficos da orientação e do erro em *x* não é coincidência. Isso acontece porque, quando seguidor e líder se deslocam no mesmo sentido e a orientação daquele é menor do que a de referência, o controlador gera um sinal para que este robô gire no sentido anti-horário, aumentando sua orientação e reduzindo a componente *x* da sua velocidade. Este processo pode ocorrer de forma inversa se a relação entre as orientações do seguidor e de referência forem contrárias. Assim, a coordenada *x* tende a reduzir com o aumento da orientação e a aumentar com a redução desta. Este fato, aliado ao atraso na atualização da postura, contribui para a semelhança entre o erro em *x* e a orientação do seguidor.

Como a componente *y* da velocidade não sofre influência direta da orientação do seguidor, o erro em *y*, embora inicialmente grande, não sofre oscilação.

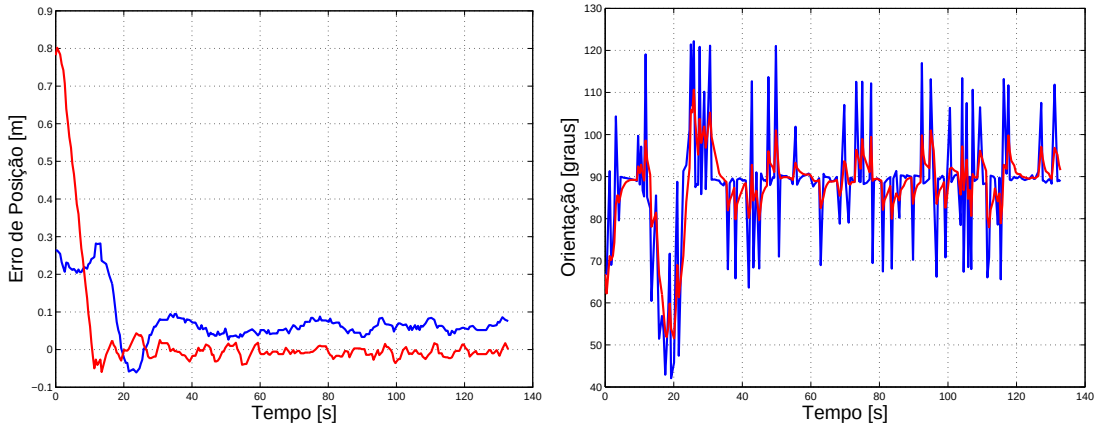


(a) (b)

Figura 4.12: Experimento 1: (a) Erros de posição e (b) Orientação do seguidor 1.

Os resultados obtidos para o seguidor 2 são mostrados na Figura 4.13. Mais uma vez, os erros de posição ficam limitados a 10 cm e a orientação em torno de 90^o. Percebe-se, porém, que o erro em *x* do seguidor 2 sofre oscilação menor que para o seguidor 1. Por outro lado, o erro em *y* apresenta uma variação maior, não ficando tão próximo de zero como ficou para o seguidor 1.

Neste ponto, vale chamar a atenção para as Figuras 4.12 - (b) e 4.13 - (b) e notar a contribuição do Filtro de Kalman: as orientações calculadas (azul), devido ao ruído nas imagens, possuem variação média de aproximadamente 20^o. Após a aplicação do filtro, ela caiu para menos de 10^o, ajudando o controlador a gerar comandos de referência mais suaves.



(a) (b)

Figura 4.13: Experimento 1: (a) Erros de posição e (b) Orientação do seguidor 2.

Observando as Figuras 4.12 - (a) e 4.13 - (a) pode-se ver que, embora os erros em x estejam limitados em 10 cm, os erros em y não passam dos 5 cm, garantindo a formação desejada. Isso pode ser comprovado através da Figura 4.14, que exibe a trajetória descrita pela equipe neste primeiro experimento.

Embora a intenção fosse descrever uma trajetória retilínea, nota-se claramente que o líder realiza uma trajetória levemente curva. Isso não é devido ao erro típico de odometria, pois o robô Pioneer 2-DX usado como líder realmente executa uma trajetória curva mesmo se sua velocidade angular for escolhida como zero. Trata-se de um defeito no eixo da roda esquerda que afeta a leitura do respectivo *encoder*, provocando um pequeno erro no sinal de velocidade que é aplicado a esta roda.

Outro detalhe importante: a leitura das velocidades linear e angular do Pioneer 2-DX, provenientes dos seus *encoders*, é (bastante) ruidosa, desencorajando sua utilização nas equações do controlador. Dessa forma, optou-se por considerar que o líder executa exatamente as velocidades que lhe foram aplicadas. Esta simplificação afeta o controlador, pois irá gerar sinais de comando usando valores de velocidades do líder que não correspondem à realidade. Mesmo assim, a equipe entra e permanece na formação desejada, mostrando o bom desempenho do controlador diante de uma situação não considerada no momento de sua concepção.

Estes inconvenientes estarão presentes em todos os experimentos, embora fiquem mais evidentes nos dois primeiros, onde a trajetória é retilínea.

Embora a leitura dos *encoders* do líder não tenha sido usada para a geração dos sinais de controle, os valores de posição foram armazenados durante os experimentos para a produção das figuras que exibem a trajetória descrita pela equipe. Uma vez conhecidas a posição absoluta

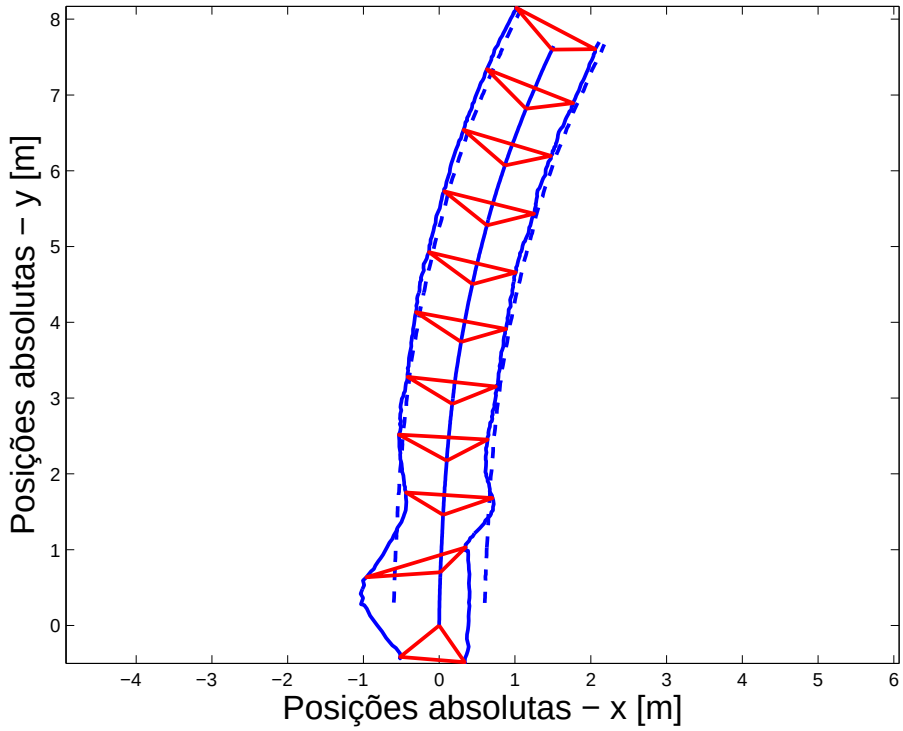


Figura 4.14: Experimento 1: trajetória descrita pela equipe.

do líder e as posições relativas dos seguidores, é possível reconstruir a trajetória de cada robô do grupo, como mostra a Figura 4.14.

4.3.2 O Experimento 2

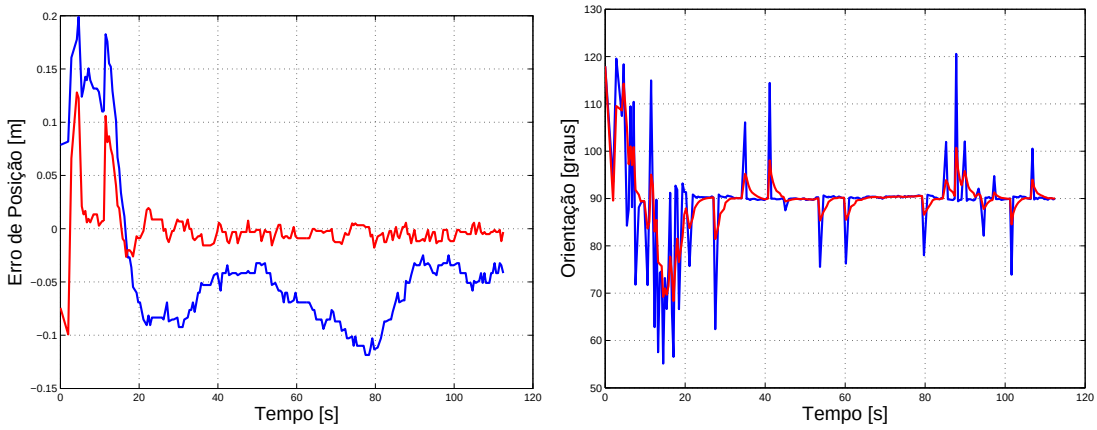
Como já foi adiantado, neste experimento o líder também desenvolveu trajetória retilínea, mas agora com velocidade de 80 mm/s e uma nova formação, com os seguidores indo atrás do líder, assim como na primeira simulação. As posições iniciais foram aproximadamente $x_{01} = -0,45$, $y_{01} = -0,4$, $x_{02} = 0,4$ e $y_{02} = -0,1$. As posições desejadas eram $x_{d1} = -0,4$, $y_{d1} = -0,5$, $x_{d2} = 0,4$ e $y_{d2} = -0,5$ e os seguidores apresentaram orientações iniciais de 120^o e 85^o. O experimento durou em torno de 130 segundos.

Como as coordenadas y iniciais eram maiores do que as desejadas, ambos os seguidores iniciam seus movimentos com velocidade linear negativa, ou seja, recuando. O seguidor 1 não tarda em inverter seu movimento, pois sua posição desejada logo passa à sua frente. O seguidor 2, por apresentar uma diferença maior entre os valores desejado e atual da coordenada y, pára e assim permanece por alguns instantes. Isso é o reflexo da soma da velocidade de formação de referência, $\dot{\xi}_{fr}$, com a velocidade de compensação do movimento do líder, $\dot{\xi}_l$ (ver Equação 2.8[2]). Estas velocidades têm sentidos opostos e o tempo em que o seguidor permaneceu parado

[2]Neste caso, $\dot{\xi}_\omega = 0$, pois a velocidade angular do líder é zero.

corresponde ao tempo onde a diferença entre elas não foi o bastante para que o controlador gerasse um sinal de comando suficiente para mover o seguidor. Este comportamento já era esperado e é uma característica positiva do controlador.

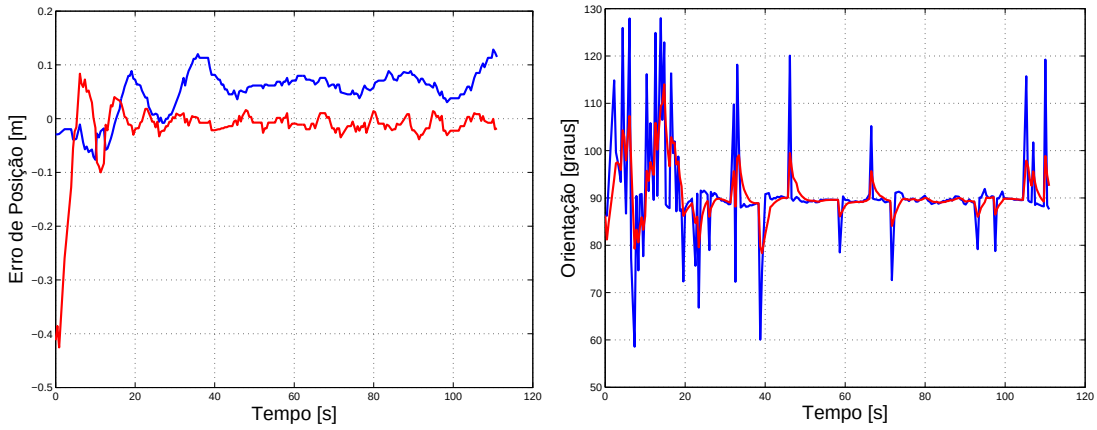
A Figura 4.15 exibe os resultados para o seguidor 1 neste experimento. É fácil notar que os erros de posição foram, desta vez, maiores que no primeiro experimento, ultrapassando os 10 cm em alguns momentos. Por outro lado, a estimativa de orientação se mostrou consideravelmente superior em relação ao caso anterior. Como a realimentação do controlador é visual, o motivo destas diferenças está na imagem. A região do espelho onde os seguidores são refletidos influencia a qualidade das estimativas de posição e orientação dos seguidores.



(a) (b)

Figura 4.15: Experimento 2: (a) Erros de posição e (b) Orientação do seguidor 1.

Para o seguidor 2, os resultados são mostrados na Figura 4.16. Assim como para o seguidor 1, os erros de posição aumentaram, mas a estimativa de orientação foi melhorada.



(a) (b)

Figura 4.16: Experimento 2: (a) Erros de posição e (b) Orientação do seguidor 2.

Outra diferença em relação ao primeiro experimento está na oscilação inicial do erro em *y*, agora presente. Entretanto, o motivo é, mais uma vez, o atraso na atualização da postura. Como os seguidores partem com velocidade linear negativa e o líder com velocidade linear positiva, as coordenadas *y* desejadas são rapidamente alcançadas, tanto que os seguidores chegam a parar. Devido ao atraso na atualização das posturas, o erro já alcança um valor tal que resulta na geração de uma velocidade linear de referência alta. Com isso, os seguidores aceleram até próximo de sua velocidade linear máxima, assim permanecendo até que a postura seja novamente atualizada. Este processo ocorre apenas no início da navegação, onde os erros iniciais são, em geral, grandes, provocando a geração de velocidades de referência com valores mais altos.

A trajetória descrita pelo grupo é exibida na Figura 4.17. Mais uma vez, esta trajetória não é bem uma reta, mas é possível ver que o grupo entra rapidamente em formação, mantendo-a até o final do experimento.

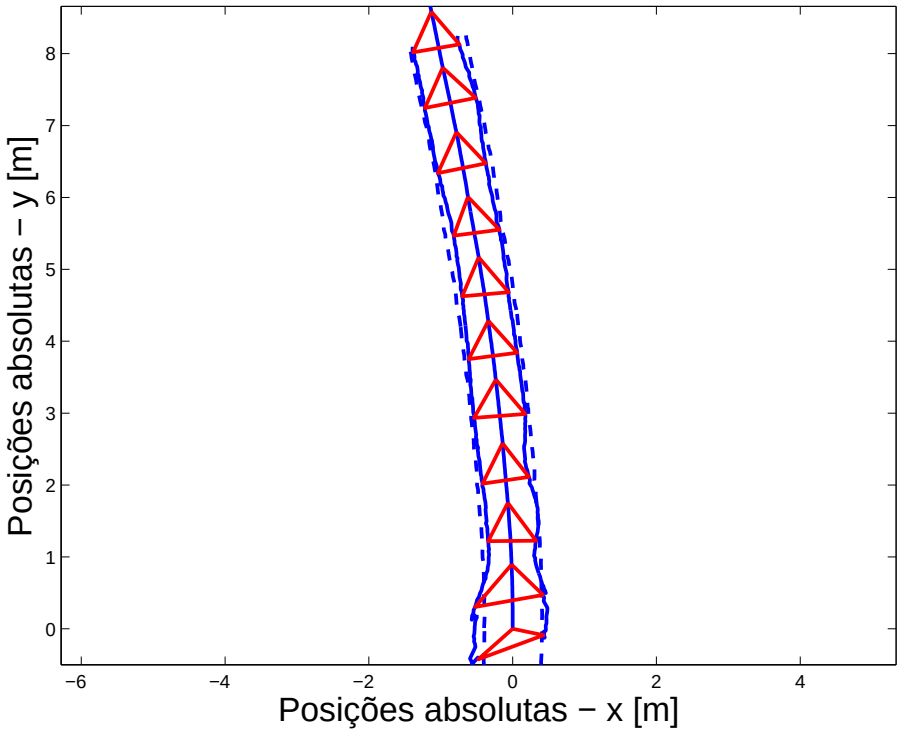


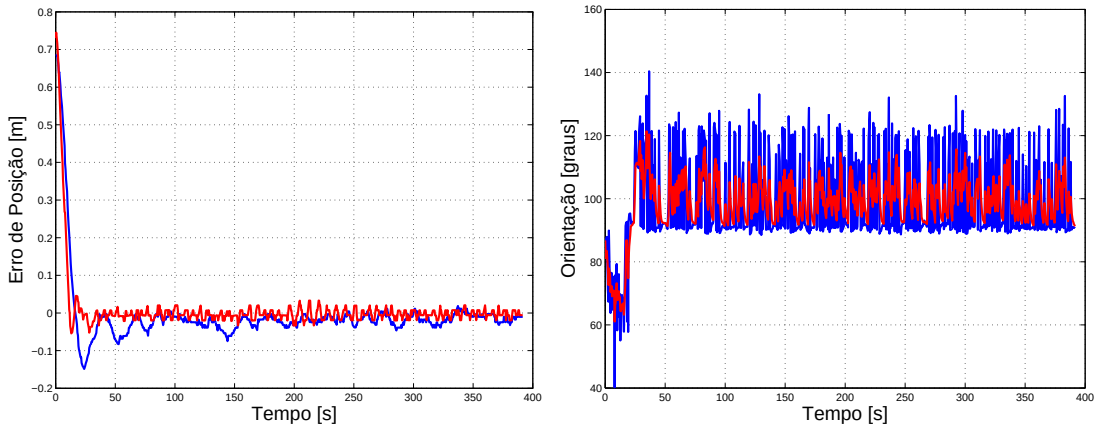
Figura 4.17: Experimento 2: trajetória descrita pela equipe.

4.3.3 O Experimento 3

Neste experimento, a trajetória descrita foi uma circunferência, pois o líder possuía velocidade linear de 60 mm/s e angular de 1,5 ^{o}/s, ambas constantes. O raio previsto para esta circunferência foi de 2,3 m, embora o observado tenha ficado maior, cerca de 3,7 m. Isso

ocorre devido ao problema no eixo da roda esquerda citado anteriormente. Os seguidores foram inicialmente detectados nas posições dadas, aproximadamente, por $x_{01} = -0,7$, $y_{01} = 0,0$, $x_{02} = 0,0$ e $y_{02} = -0,9$. Como posições desejadas, escolheu-se $x_{d1} = 0,0$, $y_{d1} = 0,7$, $x_{d2} = 0,0$ e $y_{d2} = -0,7$, ou seja, os mesmos valores usados na segunda simulação. Isso permite comparar melhor os resultados teóricos, obtidos na simulação, com os práticos, obtidos no experimento. Neste caso, os dois seguidores iniciaram a navegação com cerca de 85^o em relação ao líder.

Este experimento durou mais do que o esperado, cerca de 410 segundos (ou 6 minutos e 50 segundos), devido ao aumento no raio da trajetória descrita. Os resultados são compatíveis com aqueles obtidos na segunda simulação. Isso pode ser visto observando-se as Figuras 4.18 - (a) e 4.19 - (a) e comparando-as com as Figuras 4.9 - (a) e (b).

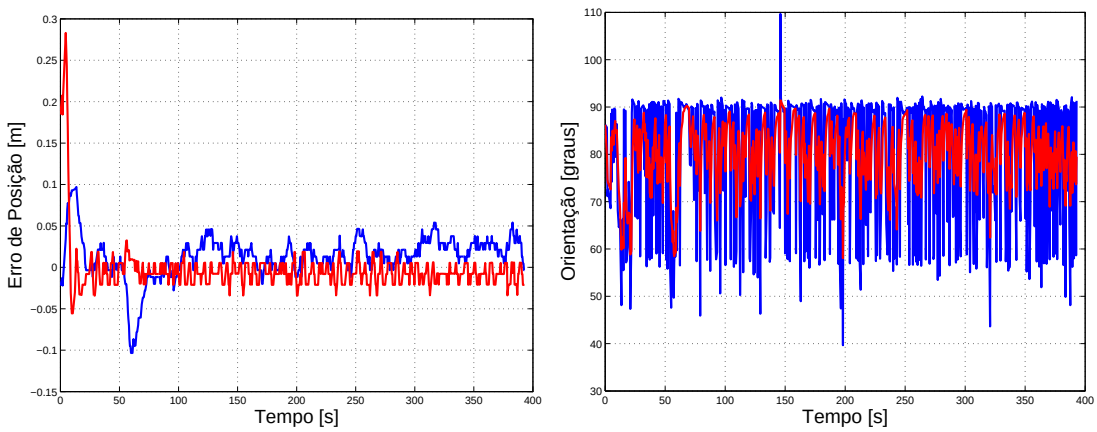


(a) (b)

Figura 4.18: Experimento 3: (a) Erros de posição e (b) Orientação do seguidor 1.

Para uma comparação mais criteriosa entre os resultados experimentais e os obtidos com simulação, foram medidos a média e o erro quadrático médio dos erros de posição de cada seguidor, tanto na simulação quanto no experimento. Essas medidas foram feitas a partir do momento em que o erro em x ou em y se torna menor do que 10 cm. Dessa forma, não são considerados os erros iniciais de posição acima de 10 cm que, além de serem normalmente grandes e prejudicarem a comparação desejada, são diferentes na simulação e no experimento.

As Tabelas 4.1 e 4.2 mostram os valores medidos. Nota-se que, para ambos os seguidores, a média obtida na primeira situação é maior, em módulo, do que a obtida no segundo caso. O erro quadrático médio relativo ao erro em x também se mostrou maior na simulação, independente do robô. Apenas o erro quadrático médio relativo ao erro em y foi maior no experimento e isso ocorre devido ao ruído de medição, facilmente observado nas Figuras 4.18 - (a) e 4.19 - (a). Vale a pena chamar a atenção para a escala da Figura 4.19 - (a), que é de 5 e não 10 cm.



(a) (b)

Figura 4.19: Experimento 3: (a) Erros de posição e (b) Orientação do seguidor 2.

		Média (cm)	Erro Quadrático Médio (cm^2)
Seguidor 1	erro x	-3,98	17,15
	erro y	-0,9	1,18
Seguidor 2	erro x	-7,42	114,1
	erro y	0,9	1,06

Tabela 4.1: Média e erro quadrático médio relativos à simulação 2.

		Média (cm)	Erro Quadrático Médio (cm^2)
Seguidor 1	erro x	-2,34	10,93
	erro y	-0,37	2,09
Seguidor 2	erro x	1,38	8,37
	erro y	-0,66	2,20

Tabela 4.2: Média e erro quadrático médio relativos ao experimento 3.

Neste ponto, é interessante fazer uma análise do pico negativo do erro em x do seguidor 2 – Figura 4.19 - (a) – quando transcorridos cerca de 60 segundos de experimento. Isso aconteceu porque o cabo de alimentação deste seguidor se prendeu momentaneamente, prejudicando seu movimento. Porém, tão logo o cabo se soltou, o controlador conduziu este seguidor de volta à formação desejada. Esse fato mostra a robustez do controle adotado, pois mesmo sofrendo a ação de perturbações mais fortes que os ruídos de medição, foi capaz de alcançar a formação desejada.

A trajetória descrita neste experimento é exibida na Figura 4.20. Como a velocidade angular do líder é positiva, o sentido do movimento da equipe é o anti-horário. Nota-se que os

robôs rapidamente entram em formação, pois o segundo triângulo desenhado (vermelho) mais se assemelha a uma reta do que a um triângulo. O mesmo vale para os triângulos seguintes, mostrando que a equipe mantém a formação até o fim do experimento. Este resultado foi o mesmo obtido na simulação 2.

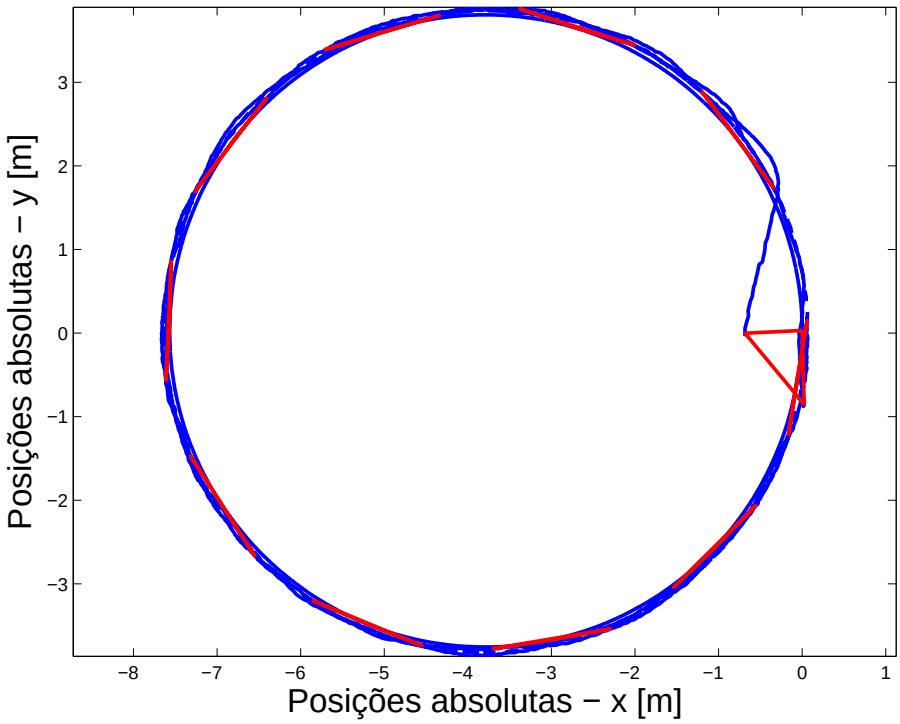


Figura 4.20: Experimento 3: trajetória descrita pela equipe.

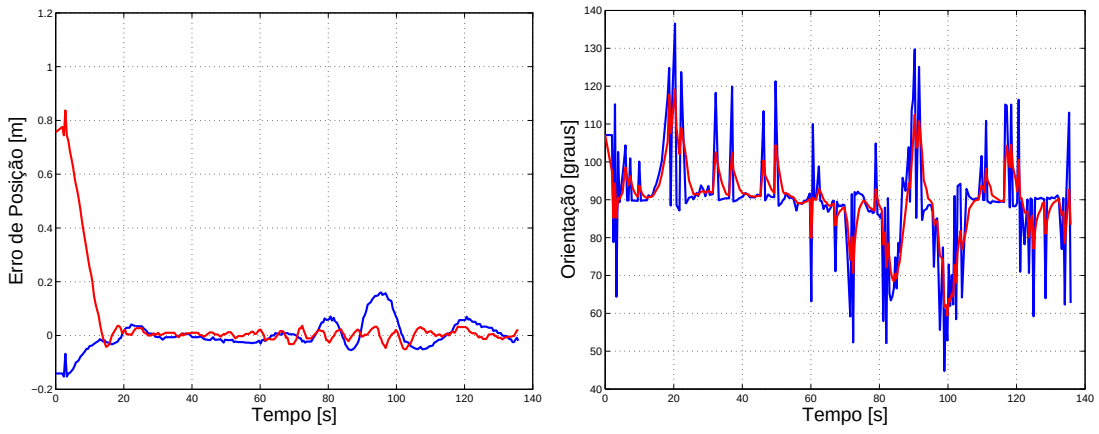
Aproveitando as comparações entre os erros de posição, pode-se fazer o mesmo entre as orientações obtidas neste experimento e na respectiva simulação: neste último caso, elas são facilmente determinadas por tornarem-se constantes assim que a equipe entra em formação, da mesma forma que ocorre com os erros de posição. As orientações em regime estacionário foram, aproximadamente, de 107^o para o seguidor 1 e 73^o para o seguidor 2, conforme Figura 4.9 - (c).

Os valores médios dos resultados experimentais ficaram em $99,68^o$ para o seguidor 1 e $80,80^o$ para o seguidor 2 (Figuras 4.18 - (b) e 4.19 - (b)). Percebe-se que, no experimento, as orientações são mais próximas de 90^o e que as diferenças entre os resultados experimentais e simulados estão em torno de 7^o, independente do robô. Essa diferença é fruto do raio (maior) descrito no experimento. Quanto maior o raio, menor será a curvatura da trajetória, fazendo com que os seguidores fiquem mais alinhados com o líder.

4.3.4 O Experimento 4

Neste experimento, o objetivo foi verificar o comportamento do controlador quando a trajetória é sinuosa, mesma situação da terceira simulação. Entretanto, há uma diferença na formação escolhida. Neste caso, os seguidores devem ir à frente do líder, com posições desejadas iguais a $x_{d1} = -0,6$, $y_{d1} = 0,3$, $x_{d2} = 0,6$ e $y_{d2} = 0,3$. As posições iniciais foram, aproximadamente, iguais a $x_{01} = -0,45$, $y_{01} = -0,45$, $x_{02} = 0,3$ e $y_{02} = -0,5$. O seguidor 1 partiu com uma orientação próxima de 105^o, enquanto o seguidor 2 iniciou seu movimento com cerca de 65^o em relação ao líder. O experimento completo durou pouco mais de 150 segundos, sendo os primeiros 20 segundos destinados à determinação das posturas iniciais dos seguidores. Assim como na simulação 3, a velocidade angular do líder foi zero nos primeiros 10 segundos de navegação, igual a $f_{\omega l}(t)$ no intervalo $10 \leq t \leq 100$s, voltando a zero nos segundos restantes.

A Figura 4.21 exibe os comportamentos dos erros de posição e da orientação do seguidor 1 durante este experimento. Nota-se que os maiores erros de posição acontecem pouco antes dos 100 segundos de navegação, onde a velocidade angular do líder está atingindo seu valor máximo, +1,5 ^{o}/s: como este seguidor está à sua esquerda, precisa aumentar a velocidade linear para manter a formação, dificultando o controle de sua postura. O mesmo não é observado no trecho onde o líder inicia o movimento de rotação, ou seja, logo após os 10 segundos de navegação, pois como o seguidor 1 está à esquerda (por dentro da curva, neste trecho), sua velocidade linear é menor, o que facilita seu controle.

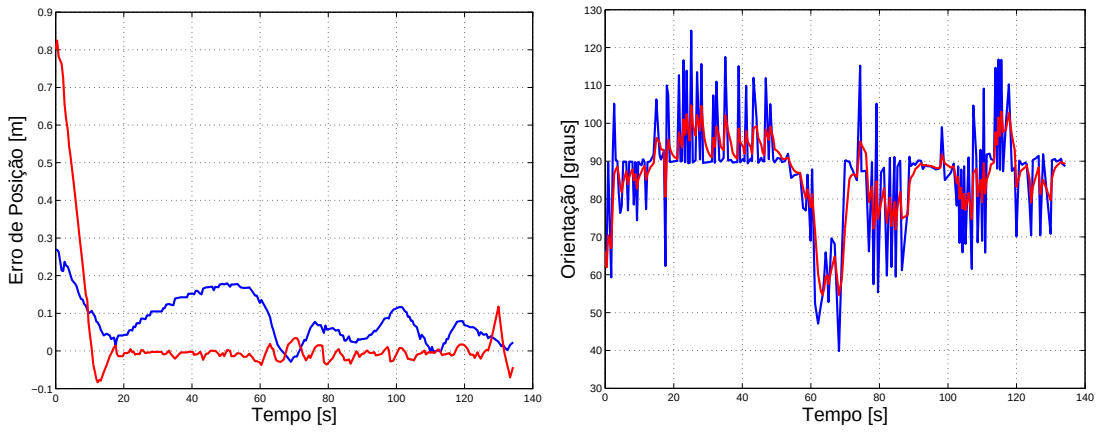


(a) (b)

Figura 4.21: Experimento 4: (a) Erros de posição e (b) Orientação do seguidor 1.

A orientação do seguidor 1 apresentou, durante este experimento, mais oscilação do que nos anteriores. Isso já foi previsto pela simulação 3 (Figura 4.11 - (c)) e ocorre devido ao tipo de trajetória executada, onde as orientações tanto dos seguidores quanto do líder estão variando no tempo.

A Figura 4.22 traz os resultados para o segundo robô seguidor. Nota-se que, como para o primeiro seguidor, os erros de posição são maiores que nos casos anteriores.



(a) (b)

Figura 4.22: Experimento 4: (a) Erros de posição e (b) Orientação do seguidor 2.

Entretanto, neste caso, os maiores erros acontecem quando o líder inicia o movimento rotatório, girando à esquerda. Como este seguidor se encontra à sua direita, deve acelerar até obter uma velocidade linear superior, o que dificulta o controle de sua postura. Assim que a velocidade angular do líder troca de sinal e este passa a girar no sentido horário, o seguidor 2 alcança a formação desejada, reduzindo seus erros de posição a valores aceitáveis.

Da mesma forma que na orientação do seguidor 1, o seguidor 2 apresenta oscilações significativas. Isso acontece pelo mesmo motivo, ou seja, devido à trajetória descrita.

A Figura 4.23 exibe esta trajetória, onde é possível perceber que o líder gira mais à direita do que à esquerda. Isso não corresponde ao desejado, uma vez que a função $f_{\omega l}(t)$ envia sinais que variam desde +1,5 ^{o}/s a -1,5 ^{o}/s, passando por zero justamente no centro do intervalo onde é válida. A razão para a execução desta trajetória, e não da desejada, está (mais uma vez) no erro de leitura dos *encoders* da roda esquerda do líder. Claramente, isso afeta o desempenho do sistema, pois o valor da velocidade angular do líder usado nas equações do controlador é exatamente aquele determinado por $f_{\omega l}(t)$, e que não corresponde ao valor real executado. Por outro lado, isso mostra que, mesmo assim, o controlador é capaz de conduzir os seguidores à formação desejada.

4.3.5 O Experimento 5

Os resultados apresentados pelo Experimento 4 foram considerados satisfatórios. Porém, decidiu-se pela realização de mais um experimento para verificar se os resultados seriam os mesmos caso os seguidores navegassem atrás do líder, como na simulação 3.

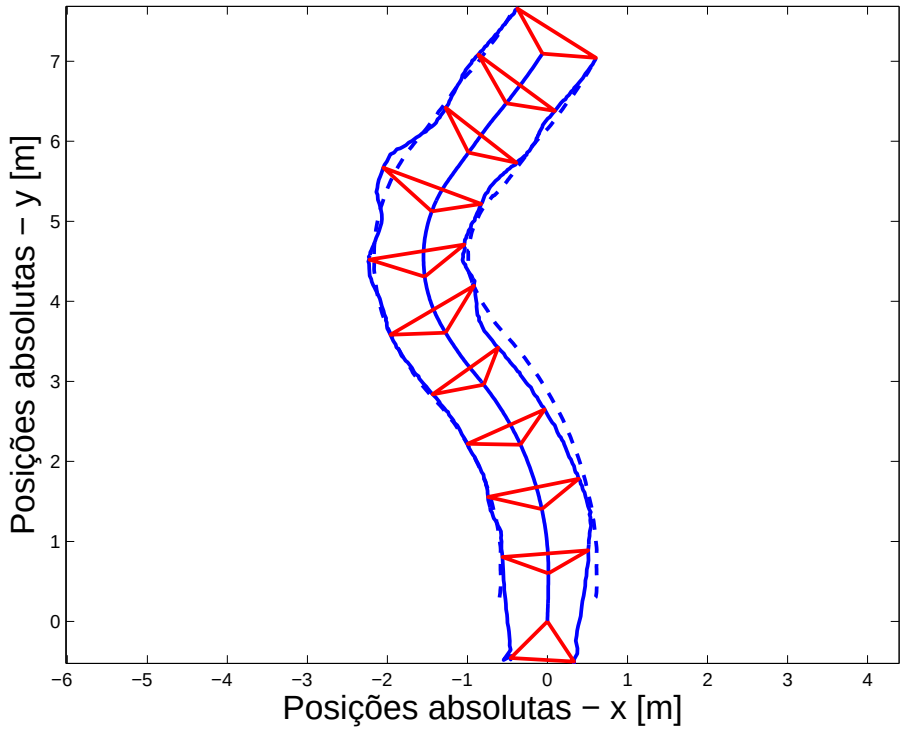
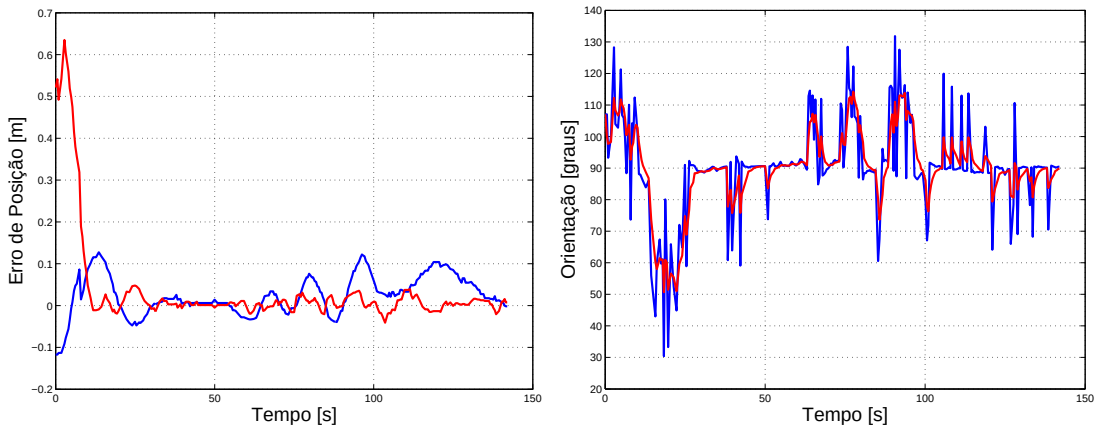


Figura 4.23: Experimento 4: trajetória descrita pela equipe.

Assim, neste experimento, as posições desejadas foram $x_{d1} = -0,5$, $y_{d1} = -0,3$, $x_{d2} = 0,5$ e $y_{d2} = -0,3$. Os seguidores foram inicialmente detectados nas posições $x_{01} = -0,4$, $y_{01} = -0,8$, $x_{02} = 0,25$ e $y_{02} = -0,85$, aproximadamente. A orientação inicial do seguidor 1 foi cerca de 105^o, enquanto o seguidor 2 partiu com uma orientação de 80^o. Este experimento durou cerca de 160 segundos, pouco mais do que o experimento anterior.

Além de observar o comportamento do controlador para uma formação diferente, outro teste também foi feito neste experimento, que consiste em verificar a robustez da detecção das posições iniciais que, como se sabe, é baseada na segmentação de movimento. Para isso, duas pessoas permaneceram caminhando próximas ao líder localizava os seguidores, com o cuidado de não obstruir a visão de nenhum deles. O resultado obtido foi a detecção correta das posições dos robôs, sem nenhum ônus para o desempenho das etapas seguintes, mostrando que esta fase do processamento de imagens é, como já foi afirmado, suficientemente robusta para o problema abordado.

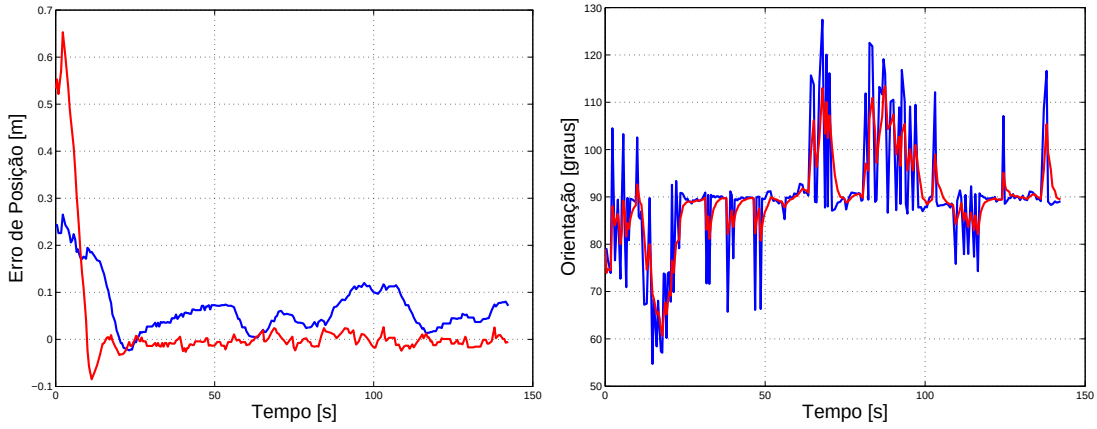
Os resultados obtidos para o seguidor 1 neste experimento são ilustrados pela Figura 4.24. É fácil perceber que a mudança na formação desejada contribuiu para reduzir os erros de posição, agora limitados em pouco mais de 10 cm. Por outro lado, a orientação do seguidor 1 não apresentou redução significativa de oscilação se observada a obtida no experimento anterior, como mostra a comparação entre as Figuras 4.21 - (b) e 4.24 - (b).



(a) (b)

Figura 4.24: Experimento 5: (a) Erros de posição e (b) Orientação do seguidor 1.

A Figura 4.25 traz os resultados obtidos para o seguidor 2. Como pode ser visto, os erros de posição deste também foram reduzidos com a alteração na formação desejada, ficando igualmente limitados em pouco mais de 10 cm. Assim como no caso do primeiro seguidor, não se observou redução significativa na oscilação de sua orientação.



(a) (b)

Figura 4.25: Experimento 5: (a) Erros de posição e (b) Orientação do seguidor 2.

Para comprovar a melhoria obtida no desempenho do controlador, basta comparar as Figuras 4.23 e 4.26. Nota-se que neste último experimento cada seguidor navega mais próximo de sua trajetória desejada, principalmente o segundo robô.

Nestes dois últimos experimentos, a equipe entra rapidamente em formação e, mesmo com as mudanças na velocidade angular do líder, a mantém até o fim. Entretanto, durante a realização destes, observou-se que no Experimento 5 os seguidores executaram movimentos mais suaves, tornando a navegação mais segura e econômica. Assim, os dois últimos resultados, juntamente com aqueles obtidos nos Experimentos 1 e 2, sugerem que o controlador tem seu desempenho levemente afetado pela formação escolhida, pricipalmente no que tange aos erros de posição.

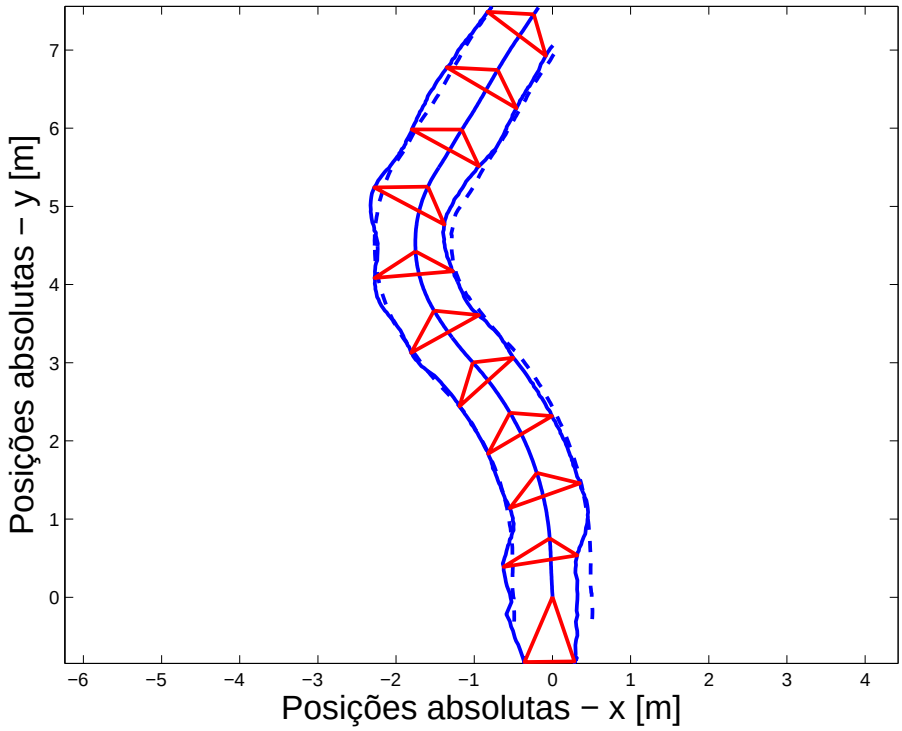


Figura 4.26: Experimento 5: trajetória descrita pela equipe.

4.4 Conclusões

Este capítulo apresentou a equipe de robôs utilizada neste trabalho, as simulações feitas e principalmente os resultados obtidos em cada um dos 5 experimentos mostrados, que foram não apenas apresentados, mas discutidos e justificados.

Como visto na simulação 3, mudanças na velocidade angular do líder têm um efeito semelhante a uma perturbação no sistema. Isso pôde ser comprovado nos Experimentos 4 e 5. Mesmo assim, a equipe entrou e manteve a formação desejada.

Os erros observados têm origem em uma série de fatores, como:

- imagem ruidosa;
- baixa resolução da câmara aliada à pequena área útil da imagem;
- trepidação do robô líder quando em movimento, que faz o sistema de visão balançar;
- baixa qualidade dos *encoders* do PIONEER 2-DX, que resulta em velocidades reais diferentes daquelas consideradas no controlador;
- o alinhamento manual, portanto, impreciso, entre o espelho, a câmara e o PIONEER 2-DX.

Como pode ser visto, a maioria destes fatores está associada ao *hardware* disponível. A aquisição de equipamentos de melhor qualidade pode reduzir significativamente estes efeitos no desempenho do sistema de controle.

Outro ponto que merece atenção é o fato de que os erros em *y* foram significativamente menores do que aqueles ocorridos em *x*, o que é mostrado em todos os gráficos apresentados. Isso faz dos erros em *x* os responsáveis pelos maiores erros registrados. A explicação para isso está no método usado para estimar a postura dos seguidores. Como apenas uma cor foi utilizada para identificar cada seguidor (devido às limitações impostas pela câmara e pelo espelho), é necessário que este apresente um deslocamento mínimo antes de ter sua postura atualizada. Dado que o movimento predominante é na direção *y* (para frente), a componente da velocidade de um seguidor nesta direção sofre menos influência da orientação à medida que entra em formação. Por outro lado, a componente *x* da velocidade de um seguidor é diretamente afetada pela sua orientação. Como a postura (e, portanto, a orientação) só é atualizada após o deslocamento mínimo, esta componente da velocidade (e, logo, o erro em *x*) torna-se mais difícil de controlar.

Observando o comportamento dos erros em *x* e *y* obtidos do ponto de vista do controlador, pode-se dizer que, devido à imprecisão das estimativas de postura dos seguidores, os sinais de controle gerados não conseguem fazer com que $\|\eta(t)\|$ e $\|\tilde{\alpha}(t)\|$ tendam exatamente a zero, como previsto na prova de estabilidade. Porém, como os erros na estimativa de postura são limitados, os erros nos sinais de controle também o são. Logo, os erros em *x*, *y* e α de cada seguidor são, também, limitados.

Em todos os casos, a equipe alcançou e manteve a formação desejada até o fim do respectivo experimento. Mesmo na presença de fatores que prejudicam o desempenho do controlador, como os cabos de alimentação e comunicação, que afetam a dinâmica dos robôs, os ruídos na imagem, a baixa resolução da câmara e a pequena área útil da imagem, os maiores erros de posição não chegaram a atingir 20 cm, ficando, na maior parte do tempo, limitados a 10 cm.

O próximo capítulo encerra esta Dissertação apresentando uma discussão geral sobre o trabalho aqui desenvolvido, assim como as principais contribuições, dificuldades e trabalhos futuros.

5 *Conclusões*

Neste trabalho, foi apresentada uma estratégia de controle de formação de um grupo de robôs móveis baseada em visão omnidirecional e controle não linear.

Uma das motivações deste projeto foi a construção de um grupo de robôs de baixo custo para a realização de tarefas através de cooperação. Como se sabe, quanto maior a capacidade de processamento e a quantidade de sensores embarcados, maior o custo associado. Isso motivou a adoção da arquitetura de controle centralizado, pois permite que o processamento e a tomada de decisão sejam concentrados em apenas um robô da equipe, aumentando o custo deste, mas reduzindo o do restante da equipe. Sem dúvida, uma arquitetura de controle descentralizado bem projetada é mais robusta que a arquitetura aqui adotada. Entretanto, isso exige que os integrantes da equipe sejam equipados com sensores e capacidade de processamento adequados, aumentando seu custo.

Assim, a equipe de robôs utilizada é composta por um líder dotado de maior capacidade de processamento e sensores. Os outros integrantes são robôs celulares equipados apenas com um microcontrolador e os dispositivos necessários para o acionamento dos motores.

O controle de formação é conseguido através da teoria de controle não linear e das técnicas de processamento digital de imagens. Para coordenar a formação do grupo, o líder utiliza um controlador não linear estável responsável por gerar os comandos necessários para que os seguidores entrem em formação e, ao mesmo tempo, acompanhem o movimento do líder. Para isso, este controlador é dividido em dois sub-controladores. O primeiro deles, chamado Controlador de Formação, foi projetado com o objetivo de conduzir os seguidores à postura desejada considerando apenas o erro de formação, desprezando as velocidades linear e angular do líder, caso existam. O segundo sub-controlador, o Controlador de Compensação, visa adicionar à velocidade de referência gerada pelo Controlador de Formação as compensações necessárias para os seguidores acompanharem o líder. A estabilidade de cada um dos sub-controladores foi provada usando o Método de Lyapunov.

A realimentação do controlador foi feita através do processamento das imagens capturadas por um sistema de visão omnidirecional. A vantagem destas imagens é que elas fornecem 360^o de campo visual horizontal e permitem que o líder estime a posição de todos os seguidores com apenas uma imagem. Neste trabalho, estas imagens foram conseguidas por meio do acoplamento de uma câmara de vídeo convencional e um espelho hiperbólico, ou seja, sem a necessidade de usar câmaras ou lentes especiais.

O processamento das imagens foi dividido em três etapas principais. A primeira delas é responsável pela detecção das posições iniciais dos seguidores. Isso é necessário porque o líder não sabe, inicialmente, onde estão os outros integrantes do grupo. A detecção da cor de cada seguidor é feita de forma automática. Logo, é possível trocar a cor de um robô ou mesmo substitui-lo por outro de cor diferente entre um experimento e outro sem a necessidade de alteração no código fonte. Técnicas de processamento digital de imagens, como *background subtraction*, operações lógicas e morfológicas, juntamente com o algoritmo de filtragem de *blobs* aqui desenvolvido, foram empregados para a implementação desta etapa.

A etapa seguinte tem a tarefa de estimar, através do uso dos algoritmos CAMSHIFT, RANSAC e Mínimos Quadrados, a orientação de cada robô seguidor para, enfim, determinar sua postura inicial.

Uma vez iniciada a navegação, entra em cena a última e mais importante etapa do processamento das imagens, que consiste no rastreamento contínuo dos robôs celulares e é responsável por fornecer ao controlador as estimativas da postura de cada um dos seguidores, ou seja, realiza a realimentação do controlador.

Para a validação do controlador proposto, simulações e experimentos com robôs reais foram realizados com o objetivo de estudar o comportamento do controlador para diferentes configurações, embora não levem em conta a dinâmica de cada robô.

Os resultados obtidos nos experimentos realizados com os robôs mostram um desempenho satisfatório para o sistema *controlador + processamento de imagens* proposto. A etapa de detecção das posições iniciais se mostrou confiável também em ambientes com iluminação irregular, piso reflexivo e movimento de pessoas. O mesmo se pode afirmar das etapas de detecção das orientações iniciais e de rastreamento para navegação. Assim como no caso das simulações, diferentes configurações foram impostas. Em todos os casos, a equipe foi capaz de entrar e manter a formação desejada. Os erros de posição ficaram, na maioria dos casos, limitados a 10 cm. Entretanto, dependendo da formação e da trajetória descrita pelo grupo, foi necessário ajustar os ganhos do controlador. Além disso, esses ganhos não são os mesmos para os dois seguidores, uma vez que estes não são idênticos.

O controlador e o processamento de imagens aqui propostos permitem que vários robôs seguidores sejam usados. A principal limitação está no campo visual do líder, já que é difícil estimar a postura de um robô muito distante. Obviamente, um número excessivo de seguidores complica o controle de formação, pois a possibilidade de colisão aumenta consideravelmente.

Este sistema pode ainda ser utilizado em aplicações que exigem uma grande quantidade (dezenas ou centenas) de robôs. Para isso, os robôs seriam divididos em várias equipes contendo, cada uma delas, um líder. Isso permite que os benefícios das arquiteturas de controle centralizado e descentralizado sejam aproveitados ao mesmo tempo. Enquanto cada equipe trabalha com controle centralizado, a coordenação entre as equipes pode ser feita através de uma abordagem de controle descentralizado. Inclusive, no caso de uma falha de um dos líderes, seus seguidores podem ser "adotados" por uma equipe vizinha, garantindo mais robustez ao sistema como um todo.

5.1 Contribuições deste Trabalho

Na seção anterior foi feita uma discussão geral sobre o desenvolvimento, a implementação e os testes do trabalho realizado nesta Dissertação de Mestrado. Apenas com o intuito de ressaltar as suas principais contribuições, as mesmas serão resumidas a seguir:

- O controlador não linear de formação baseado em visão (principal contribuição);
- As metodologias para as estimativas das posturas iniciais e das orientações de robôs celulares identificados por apenas uma cor;
- Área de trabalho ilimitada, já que o sistema de visão está embarcado no robô líder e este pode se deslocar com o grupo para diferentes locais;

É importante lembrar que, como mencionado no Capítulo 2, as primeiras equações do controlador foram propostas em [35]. Neste trabalho, elas foram adaptadas para proporcionar tanto maior controle sobre os parâmetros que influenciam o comportamento do controlador quanto a geração de sinais de comando mais suaves.

Vale ressaltar que, devido às contribuições citadas acima e aos resultados obtidos, este trabalho gerou três publicações a nível internacional e uma a nível nacional, encontradas, respectivamente, em [61, 62, 63] e [64].

5.2 Dificuldades Encontradas

Dentre todas as dificuldades encontradas, a principal foi a baixa qualidade da câmara de vídeo utilizada. Inicialmente, o sistema era composto por uma outra câmara, de outro fabricante (Gradiente, modelo SC-80), que permitia maior controle sobre os parâmetros da imagem capturada. Com isso, através do ajuste do brilho e da saturação de cores, era possível trabalhar com imagens mais escuras e cores bem saturadas. A redução do brilho conferia menor sensibilidade aos ruídos de iluminação e reflexos no piso. A alta saturação tornava o processo de segmentação de cores mais fácil e bastante estável, ou seja, as coordenadas do centróide da região segmentada apresentavam variação de apenas um ou dois *pixels*. Infelizmente, os dois modelos desta câmara disponíveis no laboratório queimaram, e passou-se a utilizar um equipamento Samsung, modelo CCS-212, que, entre outros incovenientes, varia muito pouco a saturação de cores.

A resolução da imagem capturada é de 640 x 480 *pixels*, valores modestos quando se considera que apenas uma fração do total de *pixels* é realmente utilizada para processamento. Como discutido no capítulo referente ao processamento de imagens, a forma inadequada do espelho disponível e a câmara empregada exigem a determinação de uma área de interesse que limita a visão do líder num raio de 2 metros, descartando a maior parte da área total da imagem capturada. Além disso, considerando que a região mais central da imagem é ocupada pela câmara e pelo robô líder, a área útil para processamento se torna pouco mais de 12% da área total disponível na imagem. Com isso, a estimativa de postura é penalizada, afetando o desempenho do controlador. Mais uma vez, vale lembrar que isso acontece devido à câmara e ao espelho utilizados, e não ao uso de imagens omnidirecionais.

Embora de menor importância, dois fatores associados ao robô Pioneer 2-DX (líder da equipe) também prejudicaram o desempenho do grupo. O primeiro deles é a má qualidade da leitura dos seus *encoders*, o que impossibilitou medir suas velocidades linear e angular e utilizar estas medidas nas equações do controlador. O segundo fator é um empeno no eixo da roda esquerda, fazendo o líder balançar mais do que o normal. Com isso, o sistema de visão também balança, inserindo ruído na medição da posição dos seguidores. Por fim, provavelmente devido a estes dois fatores, o líder é incapaz de navegar em linha reta, ou seja, mesmo sob o comando de seguir em frente com velocidade angular zero, o robô descreve uma trajetória circular. Isso afeta o desempenho do controlador, pois neste caso a compensação da velocidade angular não está sendo feita.

5.3 Trabalhos Futuros

A primeira e mais importante melhoria consiste em substituir o sistema de visão por um outro com uma câmara de melhor resolução e um espelho cuja forma permita visualizar melhor a região em torno do sistema. Com isso, será possível utilizar duas cores em cada seguidor e suas orientações poderão ser determinadas usando-se os centróides de cada cor. Isso não elimina a necessidade de filtragem, mas fornece resultados mais precisos.

Outro ponto de destaque são as simulações. Um ambiente virtual para a realização de simulações é uma importante ferramenta no campo da robótica, como em várias outras áreas do conhecimento. Assim, desenvolver um ambiente deste tipo levando em consideração o modelo dinâmico tanto do líder quanto dos seguidores contribuirá, sem dúvida, na tarefa de encontrar os melhores ganhos para o controlador. A partir daí, as simulações permitirão sintonizar o controlador empregado sem a necessidade de realizar experimentos com esta finalidade. Técnicas de Identificação de Sistemas podem ser usadas para obter esses modelos, que podem ser usados não apenas nas simulações, mas incorporados ao controlador. Dessa forma, os sinais de controle gerados conduzirão a equipe à formação desejada de maneira mais eficiente.

Para tornar o sistema mais flexível e robusto, pode ser implantado um algoritmo supervisor com o objetivo de ajustar automaticamente os ganhos do controlador em função do desempenho da equipe na tarefa de entrar e manter a formação desejada.

Como a utilização de vários robôs é permitida, trabalhos podem ser feitos abordando problemas como mapeamento, exploração, limpeza, resgate, vigilância e outros. Além disso, um grupo com um grande número de robôs pode ser dividido em várias equipes menores, sendo cada uma delas uma instância do trabalho apresentado nesta Dissertação. A cooperação entre as equipes pode ser coordenada através de uma arquitetura de controle descentralizado.

Como próxima etapa deste projeto pretende-se incluir módulos Zigbee para a comunicação sem fio entre os robôs da equipe. Em seguida, espera-se adicionar algoritmos para detecção e desvio de obstáculos baseados em visão (fluxo ótico e tempo para colisão), sensores de ultrasom ou mesmo sensores *laser*. Estes últimos seriam usados apenas no líder devido ao seu custo e ao seu peso. Pretende-se, ainda, aplicar a fusão dos dados provenientes das imagens omnidirecionais e do sensor *laser* para a construção de um mapa global do ambiente. Isso permitirá a realização de um *SLAM* (*Simultaneous Localization and Mapping*), onde o líder conduzirá a equipe pelo ambiente ao mesmo tempo que construirá um mapa global deste.

Por fim, uma análise da influência dos erros de medição sobre o controlador pode fornecer informações importantes para a melhoria do controle utilizado, bem como para o projeto de um novo controlador.

Referências Bibliográficas

[1] BEKEY, G. A. *Autonomous Robots: From Biological Inspiration to Implementation and Control*. [S.l.]: The MIT Press, 2005.

[2] PIO, J. L. de S.; OLIVEIRA, C. J. S. *Visão Omnidirecional*. maio 2004. www.npdi.dcc.ufmg.br/workshop/wti2004/apresentacoes/a084-pio.pdf. V Workshop em Tratamento de Imagens.

[3] CAO, Y. U.; FUKUNAGA, A. S.; KAHNG, A. B. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, v. 4, p. 7–27, 1997.

[4] SQUYRES, S. *Spirit, Opportunity and the Exploration of the Red Planet*. [S.l.]: Scribe Publications, 2005.

[5] ARAI, T.; PAGELLO, E.; PARKER, L. E. Guest editorial - advances in multirobot systems. *IEEE Transactions on Robotics and Automation*, v. 18, n. 5, Outubro 2002.

[6] WOOLDRIDGE, M. *An Introduction to MultiAgent Systems*. [S.l.]: John Wiley & Sons, LTD, 2002.

[7] FUKUDA, T.; NAKAGAWA, S. A dynamically reconficurable robotic system (concept of a system and optimal configurations). In: *IECON*. [S.l.: s.n.], 1987. p. 588–595.

[8] BENI, G. The concept of cellular robot. In: *IEEE Symposium on Intelligent Control*. [S.l.: s.n.], 1988. p. 57–61.

[9] PREMVUTI, S.; YUTA, S. Consideration on the cooperation of multiple autonomous mobile robots. In: *IEEE/RSJ IROS*. [S.l.: s.n.], 1990. p. 212–219.

[10] ARAI, T.; OGATA, H.; SUZUKI, T. Collision avoidance among multiple robots using virtual impedance. In: *IEEE/RSJ IROS*. [S.l.: s.n.], 1989.

[11] WANG, P. K. C. Navigation strategies for multiple autonomous mobile robots. In: *IEEE/RSJ IROS*. [S.l.: s.n.], 1989. p. 486–493.

[12] ASAMA, H.; MATSUMOTO, A.; ISHIDA, Y. Design of an autonomous and distributed robot system: Actress. In: *IEEE/RSJ IROS*. Tsukuba, Japan: [s.n.], 1989. p. 283–290.

[13] BALCH, T.; PARKER, L. E. *Robot Teams: from Polymorphism to Diversity*. Natick, MA: A. K. Peters 2002.

[14] SCHULTZ, A.; PARKER, L. E. *Multi-Robot Systems: From Swarms to Intelligent Automata*. Norwell, MA: Kluwer 2002.

[15] ASAMA, H. et al. *Distributed Autonomous Robotic Systems 6*. New York: Springer-Verlab 1998.

[16] LEUTH, T. *Distributed Autonomous Robotic Systems 3*. New York: Springer-Verlab 1998.

[17] PARKER, L. E.; BEKEY, G.; BARHEN, J. *Distributed Autonomous Robotic Systems 4*. New York: Springer-Verlab 2002.

[18] AUTONOMOUS Robots, v. 4, n. 1.

[19] AUTONOMOUS Robots, v. 8, n. 3.

[20] PARKER, L. E. Current research in multirobot systems. *Artif Life Robotics*, 2003.

[21] ARKIN, R. C. Integrating behavioral, perceptual and world knowledge in reactive navigation. *Robotic Autonomous Systems*, v. 6, p. 105–122, 1990.

[22] BROOKS, R. A. A robust layered control system for a mobile robot. *IEEE J. Robotics Automat.*, RA-2(1), p. 14–23, março 1986.

[23] VASSALLO, R. F. *Uso de Mapeamentos Visuomotores com Imagens Omnidirecionais para Aprendizagem por Imitação em Robótica*. Tese (Doutorado) — Universidade Federal do Espírito Santo, setembro 2004.

[24] MATARIC, M. *Interaction and Intelligent Behaviour*. Tese (Doutorado) — MIT, 1994.

[25] LEWIS, M. A.; TAN, K.-H. High precision formation control of mobile robots using virtual structures. *Autonomous Robots*, v. 4, p. 387–403, 1997.

[26] DAS, A. K. et al. A vision-based formation control framework. *IEEE Trans. Robot. Automat.*, v. 18, n. 5, p. 813–825, 2002.

[27] FEDDEMA, J. T.; LEWIS, C.; SCHOENWALD, D. A. Decentralized control of cooperative robotic vehicles: Theory and application. *IEEE Trans. Robot. Automat.*, v. 18, n. 5, p. 852–864, 2002.

[28] VIDAL, R.; SHAKERNIA, O.; SASTRY, S. Following the flock. *IEEE Robotics & Automation Magazine*, p. 14–20, 2004.

[29] BURGARD, W. et al. Collaborative multi-robot exploration. In: *ICRA*. San Francisco, CA: [s.n.], 2000. p. 476–481.

[30] JUNG, B.; SUKHATME, G. S. Cooperative tracking using mobile robots and environment-embedded, networked sensors. In: *IEEE International Symposium on Computational Intelligence in Robotics and Automation*. Alberta, Canadá: [s.n.], 2001. p. 206–211.

[31] DESAI, J. P.; OSTROWSKI, J. P.; KUMAR, V. Modeling and control of formations of nonholonomic mobile robots. *IEEE Trans. Robot. Automat.*, v. 17, n. 6, p. 905–908, 2001.

[32] BALCH, T.; ARKIN, R. C. Behavior-based formation control for multi-robot teams. *IEEE Trans. Robot. Automat.*, v. 14, p. 926–939, 1998.

[33] SANTOS-VICTOR, J. A.; CARELLI, R.; ZWAAN, S. V. Nonlinear visual control of remote cellular robots. *10th Mediterranean Conference on Control and Automation*, 2002.

[34] DE-LA-CRUZ, C.; CARELLI, R. Control centralizado de formación usando una cámara omnidireccional. *IV Jornadas Argentinas de Robótica*, novembro 2006.

[35] CARELLI, R. et al. Estrategia de control estable de formación para robots móviles. *AADECA 2006 - XX Congreso Argentino de Control Automático*, 2006.

[36] KELLY, R. et al. Control de una pandilla de robots móviles para el seguimiento de una constelación de puntos objetivo. *VI Congreso Mexicano de Robótica*, 2004.

[37] DE-LA-CRUZ, C. *Control de Formación de Robots Móviles*. Tese (Doutorado) — Facultad de Ingeniería de la Universidad Nacional de San Juan, San Juan, Argentina, novembro 2006.

[38] WEIMERSKIRCHAND, H. et al. Energy saving in flight formations. *Nature*, n. 413, p. 697–698, 2001.

[39] BROWN, R. G.; JENNINGS, J. S. A pusher/steerer model for strongly cooperative mobile robot manipulation. In: *IEEE/RSJ IROS*. [S.l.: s.n.], 1995. p. 562–568.

[40] CHAIMOWICZ, L. et al. An architecture for tightly-coupled multi-robot cooperation. In: *ICRA 2001*. [S.l.: s.n.].

[41] NAYAR, S. K. Omnidirectional vision. *8th International Symposium on Robotics Research*, 1997.

[42] BAKER, S.; NAYAR, S. K. A theory of single-viewpoint catadioptric image formation. *Int. Journal of Computer Vision*, v. 35, n. 2, p. 1–22, 1999.

[43] PARKER, L. E. Alliance: An architecture for fault-tolerant, cooperative control of heterogeneous mobile robots. In: *IROS 1994*. [S.l.: s.n.].

[44] GUSTAFSON david A.; MATSON, E. Taxonomy of cooperative robotic systems. In: *IEEE International Conference on Systems, Man and Cybernetics*. [S.l.: s.n.], 2003. v. 2, p. 1141–1146.

[45] BARNES, D.; GRAY, J. Behaviour synthesis for co-operant mobile robot control. In: *International Conference on Control*. [S.l.: s.n.], 1991. p. 1135–1140.

[46] VIDYASAGAR, M. *Nonlinear Systems Analysis (Classics in Applied Mathematics)*. Second. [S.l.]: SIAM, 2002.

[47] SCHNEIDER, E. et al. Eye movement driven head-mounted camera: It looks where the eyes look. In: *IEEE International Conference on Systems, Man and Cybernetics*. Hawaii: [s.n.], 2005. v. 3, p. 2437– 2442.

[48] KIM, S. I. et al. A fast center of pupil detection algorithm for vog-based eye movement tracking. In: *IEEE Engineering in Medicine and Biology 27th Annual Conference*. Shanghai, China: [s.n.], 2005. p. 3188–3191.

[49] SORIA, C. M. et al. Control de robots celulares en base a visión artificial. In: *Anais do VI Simpósio Brasileiro de Automação Inteligente - VI SBAI*. [S.l.: s.n.], 2003.

[50] OPEN Source Computer Vision Library. At http://www.intel.com/technology/ computing/opencv/index.htm.

[51] VASSALLO, R. F. et al. Bird's eye view remapping and path following based on omnidirectional vision. *XV CBA*, 2004.

[52] PEREIRA, F. G. et al. Calibração de sistemas catadióptricos e detecção da pose de robôs móveis por segmentação de imagens omnidirecionais. *VII SBAI*, 2005.

[53] GONZALEZ, R. C.; WOODS, R. E. *Digital Image Processing*. [S.l.]: Prentice Hall, 2002.

[54] BRAGANÇA, J. de O. *Estratégias para Deslocamento de Cargas Através Cooperação Robôs Móveis a Rodas*. Dissertação (Mestrado) — Universidade Federal do Espírito Santo, Vitória, ES, dezembro 2004.

[55] FISCHLER, M. A.; BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. In: FOLEY, J. D. (Ed.). *Communications of the Association for Computing Machinery (CACM)*. [S.l.: s.n.], 1981. v. 24, n. 6, p. 381–395.

[56] INC., M. R. Http://www.mobilerobots.com/.

[57] TEXAS INSTRUMENTS. *MSP430x1xx Family - User's Guide*. Disponível em http://www.ti.com.

[58] TEXAS INSTRUMENTS. *MSP430x13x, MSP430x14x Mixed Signal Microcontroller*. Disponível em http://www.ti.com.

[59] ZAMBON, E. Uma plataforma para desenvolvimento de robôs de baixo custo com alta capacidade computacional - irobot. *Universidade Federal do Espírito Santo*, Abril 2004.

[60] Sá, F. B. de. *Cooperação de Robôs Baseada em Visão Omnidirecional*. [S.l.], 2007. Projeto de PIBIC 2005/2006.

[61] GAVA, C. C. et al. A nonlinear control applied to team formation based on omnidirectional vision. In: *International Symposium on Industrial Eletronics - ISIE*. Montréal, Canadá: [s.n.], 2006.

[62] GAVA, C. C. et al. Team formation based on nonlinear control techniques and omnidirectional vision. In: *First IFAC Workshop on Multivehicle Systems - MVS*. Salvador, Brasil: [s.n.], 2006.

[63] GAVA, C. C. et al. Nonlinear control techniques and omnidirectional vision for team formation on cooperative robotics. In: *IEEE International Conference on Robotics and Automation- ICRA*. Roma, Itália: [s.n.], 2007.

[64] GAVA, C. C. et al. Controle de formação de uma equipe de robôs baseado em técnicas não lineares e visão omnidirecional. In: *VIII Simpósio Brasileiro de Automação Inteligente - SBAI*. Florianópolis, Brasil: [s.n.], 2007.